Bilkent University

Department of Computer Engineering

Object-Oriented Software Engineering Term Project

TrackIn: Intern Tracking System

Final Report

**Group 3G:**

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

**Contents:**

1. Implementation Flow
2. Changes in the Design
3. User Guide
4. Conclusion
5. References
6. **Implementation Flow**

In order to successfully achieve the implementation of TrackIn in an efficient and fair way we decided to equally divide the workload between the team members, by assigning the development of one of the subsystems to every member. We all worked simultaneously on the development since the challenges that we faced were similar and we could apply common solutions. As our general application has Client-Server architecture and MVC pattern is used in the server side ( JavaServer Faces), we first decided to separate work by the View, Controller, and Model classes. Yet, the separation of the work was evolved to implementing the View, Controller and Model together for each functionality of the subsystems.

During the implementation, we were using TomCat server for our servlet implementation in the first iteration but after we learn more about the JavaServer Faces Framework and Java EE environment, some of the techniques were deprecated (e.g. ManagedBean annotations for classes) and we need to alter them according to the modern approach. However, TomCat server implementation doesn’t allow these approaches such as CDI (Contexts and Dependency Injection) and we had to change our server implementation to Glassfish. During the implementation, Glassfish server also became insufficient for our needs and started to define unexpectedly ( Some known issues regarding jdk version). Thus, we have changed our servlet to TomEE which meet our need through the implementation.

We have encountered many difficulties when re-configuring the project on our local machines and importing the projects from github. Most of these difficulties were about the IDE that we have chosen ( Intelij IDEA). We searched about project management techniques and tools, making the project IDE independent. As a result, we started to use Maven to achieve this purpose and added the framework to our project. We also contemplated about changing our IDE to Eclipse. Yet, it was hard to change since we were inexperienced for Eclipse IDE and it was a critical time. In addition, we used Slack as our communication channel for both flexible communication when we worked remotely and for file/link sharing.

**2. Changes In The Design**

Generally, in the implementation of our project we followed the same design as described in the previously prepared reports since they were able to support the functionality of all the features our application intended. However several changes were made along the way In order for the implementation to be as efficient as possible.

* One of the greatest challenges we faced during the implementation was to find the appropriate Web Server that completed all the requirements and needs of our project. We firstly started with *Tomcat* but after many trials we understood that it did not support JSF which was crucial to our implementation. After that we tried GlassFish Web Server which did support JSF, however we faced several incompatibility problems because of the JDK version which we could not find access to. Ultimately we settled on TomEE Web Server which was supporting JFS and was compatible with our versions of JDK.
* Another change made was the addition of some classes that helped our implementation (Uitility classes such as ServletConfigListener).
* The package structure was altered during the implementation phase and to facilitate the implementation several packages were added. We removed the initial View and Controller packages and instead we added Bin, Util ,Internship and Task Management packages.
* At the very first implementation process of the project, we decided and planned our model as in a way that Internship class will be the main controller in the Internship subsystem. It was planned to play the role of the managed bean in our implementation so that the control of the objects will be based upon Internship managed bean class. However, as it went so complex in terms of the design of the subsystem, we decided to find another way around in the process of the implementation of the subsystem. Therefore, we decided to separately distribute the methods in the Internship managed bean to the other entity classes and consider all of these classes as the managed beans. As time went on with the implementation of the managed bean and its relations with the other classes, we figured out a new way which was very similar to the way that we have tried for the first trial of our implementation.

We created one managed bean for every subsystem of ours, i.e. Task Management, Internship and Account managed beans. All the other classes are considered as the entity classes in which the managed beans are using the instances of them and by means of the getter and setter methods of the entity classes, the managed beans are able to write data, get and set attributes and do the functional parts of the system via the calls that are included in the xhtml files. Therefore, by means of that design, especially the Internship subsystem, the subsystems are well organized and they are much likely to be well structured in the process of the communication with the MySql database. The connection of the database is done only once in the connector class, but the instance of this connection is used inside the methods of the managed beans. Thus, by means of the methods such as *createInternData*, *createSupervisorData*, *signUp*, *signIn*, etc. the communication is provided throughout managef beans such as Internship and the data is written to or collected from the database tables inside the database in order to be used in some functionalities like comparisons, data manipulations, calculations, etc.

* Starting from the initial design changes were made to the hierarchy of the object design and some classes were merged with others since they supported the same functionalities. As an example, in the Task Management Subsystem, “*Internship Class”* subclass is merged in the “*Task”* class, making the code more compact and functional.

**3.User Guide**

* The users of the Track-in are consist of three types: Company, Intern,and Supervisor.

The path of navigation is very similar for interns and supervisors. They both need to sign up to the system and they become free intern and free supervisors. They need to have a unique key to join a company. These unique keys can be distributed by company through e-mails or any mean of communication.

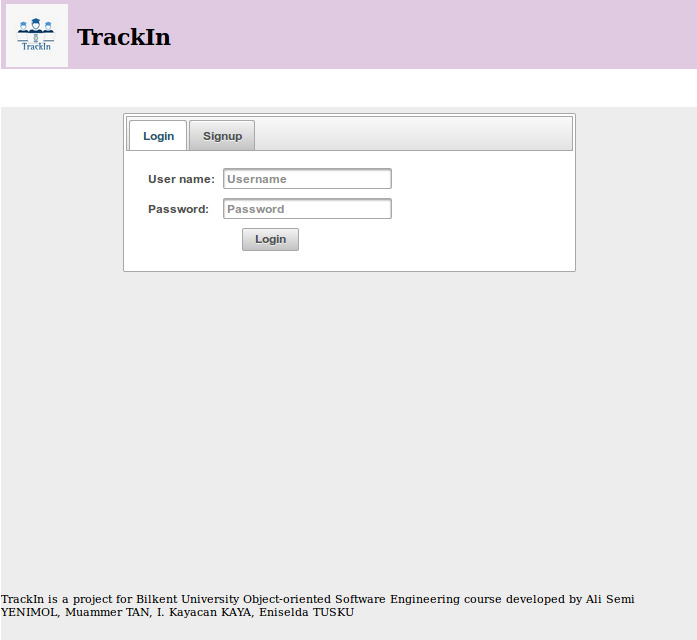


Figure Track-In HomePage- LogIn

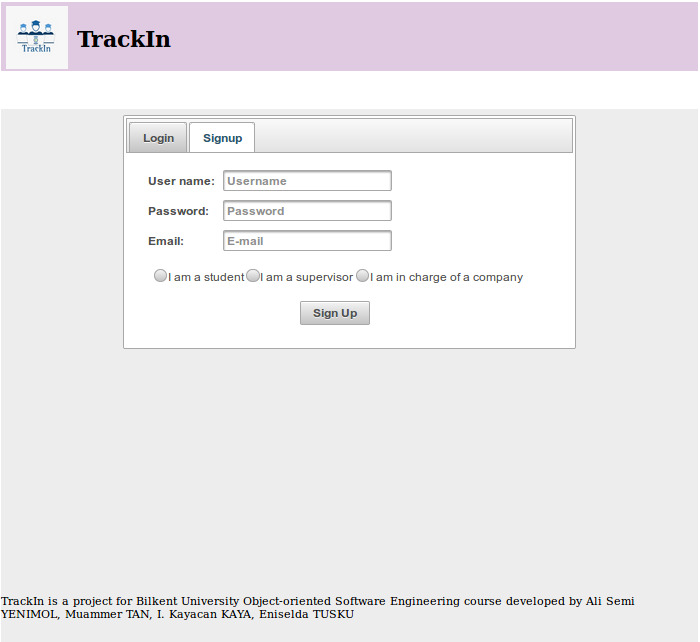


Figure Track-In Homepage - Signup

* When an intern has joined to a company, they have 5 option to do: They can see their tasks and complete them, they can see the general events in the company, they can see their schedule, they can see their documents that they upload or company provides, and they can their profile page where they can edit their credentials and information.
* After supervisors joined to a company, they can see their individual interns and the teams that supervisor creates and manages, they can see the tasks they gave and the situation of them where they can also approve the completed tasks or reject them. Similar to the intern, they can see their schedule and their profile page that has the same functionality with the interns’.

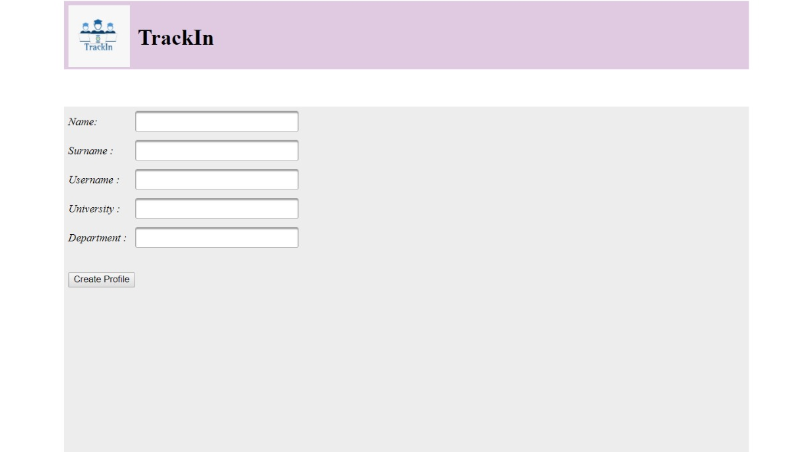


Figure Sample Profile Creation

* Company user type is expected to be controlled by Human Resources departments of the companies. They have 5 pages: they can see their registered interns and give them individual tasks without supervisor (including document/information requests), they can see their registered supervisors, they can see the events and create/control them, they can do the matching of interns to supervisors in another page, and they can see their profile and edit it.

**4.Conclusions**

* The design and development of this project was of a great significance to all of us since it introduced us to a whole new range of information and practices that helped us gain several hard and soft skills. Some of the things we believe to have obtained from all this experience are as follows:
* This project helped us to learn how to work, share ideas and integrate in a group with people we did not know before (since the groups were assigned randomly). This taught us a lot about communication, work division and integration of several ideas and individual work in a single unit.
* Being able to create and document all the phases of the development of a project with the process of analysis, design and implementation was very challenging and important and gave us a glimpse of how projects in real life working environments are supposed to work and what is expected from us as future engineers. We all believe that developing this project put things into perspective for us.
* We gained a lot of technical skills since we dealt with new working environments such as different types of Web Browsers, but also were challenged in the ones we had previous knowledge such a database and Java programming language. We strongly believe these software development skills will help us be more prepared for other projects in the future.
* We were introduced to new platforms for project management such as GIT.
* We learned to validate and examine a project not only in terms of functionality like we were generally used to but rather to see all the features such as extendibility, security, response time etc, and we understood that a good project should cover all of them.
* We managed to obtain a functional program which completed in a major extend all the functionalities that we anticipated in our reports. This was achieved with common work and ultimately we are very proud of our work as a team.