Université Antonine
Faculté d'Ingénieurs en Informatique et Télécommunications
كلية هندسة الاتصالات والمعلوماتية

Lab. Software Design
Academic Year 2023-2024
Project Deliverable

1- TEAM

First name: Hassan
Last name: Khadra
e-mail: 202110272@ua.edu.lb

2- DESCRIPTION OF THE PROJECT [10 lines]

The Software Design Project focuses on creating an Internship Program System, which can be web-based, mobile, or both. This system simplifies the management of internships by enabling interns to find opportunities in a company, apply, and track their progress. It also assists the organization/company in posting opportunities, managing applications, and evaluating interns. The main actors are interns and administrators/mentors.

Key use cases to consider include the intern's ability to apply for opportunities, track application status in real-time, and access valuable resources, as well as the administrator's capacity to efficiently handle incoming applications, assess intern performance, and mentor interns who are potential employees.

PART I

3- USE CASE DIAGRAM

In the prepared use case diagram, I have created 2 actors, Intern and Mentor/Administrator. The intern has the capability to apply for internship programs and track the application status after authenticating his account with either mobile or email and connect to the company/organization services to verify and validate him. The intern can then be rejected or accepted by the mentor. In case of acceptance, he can access learning resources or communicate with his mentor and so on (main use cases).
The mentor has the capability to view the intern application and download/view their CV to accept or reject them, he can also manage internship programs and communicate with his interns as well as view their progress and so on.

4- TEXTUAL DESCRIPTION OF USE CASES

1 - **Use Case Name:** Communicate with Mentor
Actor(s): Intern (primary), Mentor/Administrator (secondary)
Summary Description: Enables interns to engage in direct communication with their assigned mentors or administrators within the Internship Program System.
Pre-Condition:
- The intern has successfully logged into the Internship Program System.
- The mentor/administrator is available and connected to the system.
Post-Condition(s):
- The intern has sent a message or scheduled a meeting with the mentor/administrator.
Basic Path:
1. The intern logs into their account in the Internship Program System.
2. The system verifies the intern's credentials and authenticates their access.
3. Upon successful authentication, the intern accesses the communication interface.
4. The intern selects the option to communicate with the mentor/administrator.
5. The system presents options for messaging or scheduling a meeting.
6. The intern selects either to send a message or schedule a meeting.
7. If messaging:
   - The intern drafts and sends a message to the mentor/administrator.
   - The system confirms successful message delivery.
8. If scheduling a meeting:
   - The intern selects a preferred date and time for the meeting.
   - The system checks the mentor/administrator's availability.
   - If available, the system confirms the meeting schedule and notifies both parties.
9. The mentor/administrator receives the message or meeting invitation.
10. If a meeting is scheduled, both parties attend the meeting at the designated date and time.
Alternative Paths:
2a. Invalid credentials or failed authentication.
5a. Mentor/administrator is temporarily unavailable online.
7a. Message delivery failure (due to technical issues or network problems).
8a. Mentor/administrator is unavailable at the proposed meeting time.
8b. Conflicting schedules between intern and mentor/administrator.
10a. Intern or mentor/administrator fails to attend the scheduled meeting.

2 - **Use Case Name:** Apply for Program
Actor(s): Intern (primary), Internship Program System (secondary)
Summary Description: Allows interns to apply for a specific internship program within the Internship Program System.
Pre-Condition:
- The intern is authenticated and logged into the system.
- The internship program is available and open for applications.
Post-Condition(s):
- The intern's application is submitted to the program.
- The intern receives confirmation of application submission.

Basic Path:
1. The intern navigates to the list of available internship programs.
2. The system displays a list of open programs.
3. The intern selects a specific program to apply for.
4. The system presents details and requirements for the selected program.
5. The intern reviews the program details and confirms eligibility.
6. The system prompts the intern to upload their CV or relevant documents.
7. The intern uploads their CV or required documents.
8. The system validates the uploaded documents.
9. The intern confirms the application submission.
10. The system records the application and sends a confirmation to the intern.
Alternative Paths:
2a. No open programs available.
5a. Intern does not meet program requirements.
6a. Invalid document format or size.
8a. Document validation failure.
9a. Intern cancels application submission.

3 - **Use Case Name:** Provide Feedback
Actor(s): Intern (primary), Administrator/Mentor (secondary)
Summary Description: Enables interns to offer structured feedback regarding their internship experience to the company/organization.
Pre-Condition:
- The intern is authenticated and logged into the Internship Program System.
- The internship period or a significant phase of the program is completed, providing grounds for feedback.
Post-Condition(s):
- The feedback provided by the intern is successfully submitted and recorded in the system.
- The company/organization acknowledges receipt of the intern's feedback and may take actions based on the provided insights for program enhancement.
Basic Path:
1. The intern logs into the Internship Program System using valid credentials.
2. The system confirms the intern's access and identifies completed or ongoing internship phases.
3. The intern navigates to the feedback section within the system.
4. The system presents options for providing feedback, such as rating various aspects of the program and leaving textual comments.
5. The intern selects the appropriate feedback options and enters comments if required.
6. The intern submits the feedback.
7. The system acknowledges successful submission and records the feedback in the database.
8. The company/organization reviews and acknowledges receipt of the feedback.
Alternative Paths:

2a. Intern not yet eligible to provide feedback due to incomplete internship phases.
4a. Insufficient options for providing feedback.
5a. Incomplete feedback.
8a. Company/organization does not acknowledge feedback.

Further details:

Intern Primary Use Cases:

1. **Track Application Status**: Interns can monitor the real-time status of their internship applications. This stand-alone use case allows them to stay informed about the progress of their applications.
2. **Access Learning Resources**: Interns can access valuable learning materials and resources. This extends their capabilities, enabling them to upload their CV for a program application and view the programs they have been accepted into. This access extends their ability to track progress, attend meetings, download materials, and take assessment exams.
3. **Communicate with Mentor**: Interns have the ability to communicate with their mentors. This use case extends "Request Feedback" and "Schedule Meeting," facilitating open communication and feedback exchange with mentors.
4. **Provide Feedback**: This use case allows interns to provide structured feedback to the company/organization, offering insights for program improvement.

Mentor/Administrator Primary Use Cases:

1. **Manage Internship Applications**: Mentors are responsible for managing internship applications. This use case encompasses actions such as viewing interns' CVs which is an extended ability, and accepting or rejecting them, streamlining the application evaluation process.
2. **Manage Internship Programs**: Mentors can manage internship programs. This use case includes the ability to post, delete, or modify internship programs that are extended from the "Manage Internship Programs" use case. In addition, set assessment exams, post materials, and schedule meetings are functions that are extended from the "Modify Program" use case, ensuring program flexibility.
3. **Communicate with Intern**: Mentors can communicate with interns, providing guidance and support. This use case extends "Provide Feedback" and "Schedule Meeting," allowing personalized interactions and feedback exchange, which is distinct from program-wide interactions.
4. **View Intern's Progress**: Mentors can track and monitor the progress of individual interns. This use case is a standalone feature, providing insights into intern performance and program success.

5- GLOBAL ARCHITECTURE OF THE SYSTEM

1. **Web Application**: The web application is one of the core platforms for the Internship Program System. It serves as the primary interface for both interns and administrators/mentors. Interns can access the application through standard web browsers, where they can search for internship opportunities, apply, and track the status of their applications. Administrators/mentors can manage applications, evaluate intern performance, and communicate with interns through the same web-based portal. This platform offers a centralized and user-friendly solution for all system functionalities.

2. **Mobile Application**: Complementing the web application, a dedicated mobile application is provided for users who prefer mobile devices. Interns and mentors can download and install the mobile app to access the same features available on the web platform. For instance, interns can apply for internships, monitor application progress, and engage with mentors conveniently on the go. This mobile application ensures a responsive and mobile-optimized user experience.

Both the web and mobile applications feature email or phone authentication for secure user account verification, and they connect to the company/organization services to validate users' information. And of course, real-time application tracking, access to learning resources, and seamless communication functionalities are consistent across both platforms.

PART II

1- CLASS DIAGRAM

The following is a brief description of some of the classes in my **Intern class diagram**:

1. **Intern:** This class represents the user intern which has attributes like ID, email, name, password, and one static attribute called LearningResource, this attribute is static because it is shared among all interns. The Intern cannot access the application functionalities without authenticating and thus the AuthenticationMethod class is connected as one-to-one with Intern class and with composition on the Intern's side (parent). The Intern can access LearningResource class which contains the array of InternshiPrograms to access the programs and either apply, or perform functionalities like download materials. The Intern class is connected to the SystemFeedback class which allows an intern to provide a system feedback for the developers, the connection is one-to-many because the intern can provide many feedbacks and it is of composition type because the feedback depends on the Intern's existence I chose this for the application's optimization. The Intern class is also connected to the Communication class which allows an intern to communicate with their mentor or the other way around by scheduling a meeting or by providing a feedback and thus the relationship is one-to-many communications on the intern's side with composition because of optimization.

2. **AuthenticationMethod:** This class has type, details and authenticated as attributes because it serves both as the authentication method identifier class and the authentication class. The class consists of methods that allow it to perform the required verification processes. For instance, it has the authenticateUser method that takes an email and a password to verify a certain user, it also has the signup methods for both Intern and Mentor classes, the mentorRegistration method allows a new mentor to fill a registration form with his information in order to later on be accepted or rejected by an admin.

3. **LearningResource:** This class contains the array of InternshiPrograms which is derived from the connection with the InternshipProgram class, this relationship is one-to-many because a LearningResource class has many programs and it is composition because if LearningResource ceased to exist, all programs will too and thus the life dependancy. This class contains the neccessary methods to manage the array of programs.

4. **InternshipProgram:** This class has a ProgramID, ProgramName, an array list of interns a start date and an end date, ProgramMentor, and an array of Internship applications as attributes. This class is connected to the InternshipApplication class with one-to-many cardinality and composition on the parent's class, (this class), the relationship between these two classes is like so because a life dependacy exists between them and because an internship program can typically possess many internship applications.

5. **InternshipApplication:** This class has an applicationID, a program and the rest of the necessary details for the intern to apply as attributes. This class allows an intern to have an internship application uploaded for a specific program and then later on judged by a mentor of the system. This is why the class has the necessary methods that allow the mentor to accept or reject an application or even download a CV.

6. **Communication:** This is the class that allows the system users to communicate with each other, it is why the class has children classes, Feedback and Meeting, the Meeting class is connected to MeetingLocation class via composition and one-to-one cardinality because they have a life dependacy and because a meeting can only have one meeting location. The communication class has protected methods that are accessed by both Feedback and Meeting classes and each of those classes has the needed methods to perform the specified communication. For instance, the newInternRequest allows the newly created feedback by the method requestFeedback in the Intern class to be saved and uploaded. In addition, a feedback needs an attribute that identifies if it is a positive feedback to help filtering the feedbacks in the applications later on, this helps scalability.

   Note : I have provided an additional Class Diagram in the 'Class Diagrams' folder that contains both Intern and Mentor classes together, basically the full application.

6- SEQUENCE DIAGRAM

The first sequence diagram shows the method applyForProgram that takes internshipProgram as parameter which will be used to apply the intern to it using the required methods in the classes Intern, InternshipProgram, and InternshipApplication.

The second sequence diagram shows the method scheduleMeeting which the intern uses to schedule a meeting between them and their personal mentor, this method takes meeting information such as startTime, endTime, location, and the mentorID. This method needs the Intern class and the Meeting class to schedule a meeting only if the timing is available and the location is available. The database is added for more accuracy.

7- DESIGN PATTERNS

I have applied the **Factory Design Pattern** in my class diagrams for the communication classe which is implemented by Feedback and Meeting.