4 Chapter 4: System Design

4.1 Introduction

This chapter presents the design of the proposed Wesal system. The design is based on the functional and non-functional requirements discussed in Chapter 3. It explains the general architecture of the system, the main classes and their relationships, the interaction between users and the system, the main cooperative training workflow, the database structure, and the user interface design principles. The system is designed to support four user roles: the student, the training organization, the university supervisor, and the administrator. Each role has different permissions according to its responsibilities in the cooperative training process. The design aims to keep the system clear, organized, and suitable for implementation using PHP, MySQL, HTML, CSS, JavaScript, and Bootstrap.

4.2 Architectural Design

Wesal is designed as a web-based application with a simple three-part structure. The first part is the web interface, which allows users to access the system through pages and forms. The second part is the PHP application, which handles the main system rules and user operations. The third part is the MySQL database, which stores the information created and used by the system.

The web interface is developed using HTML, CSS, JavaScript, and Bootstrap. It provides different pages for students, training organizations, university supervisors, and administrators. The PHP application manages authentication, role-based permissions, training opportunities, applications, training records, reports, evaluations, and recommendation matching. The database stores user accounts, profiles, opportunities, applications, training records, reports, and evaluations.

The system also supports file uploading for student resumes and training reports. The uploaded files are managed by the PHP application, while their paths and related information are stored in the database. The recommendation function is implemented as a simple matching process based on the student's major and skills and the requirements of available training opportunities. This design keeps the system simple and avoids unnecessary external services.

[Insert Figure 4.1 here]

Figure 4.1: Wesal System Architecture

As shown in Figure 4.1, users access Wesal through the web interface. Their requests are processed by the PHP application, which checks the user's role and performs the required operation. The application then reads or updates the appropriate information in the MySQL

database. This structure separates the interface, system logic, and data storage, which makes the system easier to understand and maintain.

4.3 Object-Oriented Design

The object-oriented design describes the main objects involved in the Wesal system. These objects represent the users, profiles, training opportunities, applications, training records, reports, evaluations, and recommendation process. The design includes only the main classes that are necessary to explain the system. Technical classes related to database connections, page controllers, or individual PHP files are not included because they are implementation details.

The user account is represented by the `User` class. The user's role determines whether the account belongs to a student, a training organization, a university supervisor, or an administrator. The student and organization information is represented through profile classes. The remaining classes represent the main activities of the cooperative training process.

4.3.1 Class Diagram

The class diagram shows the main classes of Wesal and the relationships between them. The `User` class stores the common account information and the user's role. A student is connected to a `StudentProfile`, while a training organization is connected to an `OrganizationProfile`. The organization profile can publish many training opportunities, and a student profile can submit many applications.

An application connects a student with a training opportunity. When an application is accepted, it can create one training record. A training record can contain several reports and can have one evaluation from the training organization. The `Skill` class is used to represent student skills and support the recommendation process. The `RecommendationService` class calculates the matching score between a student's information and available training opportunities.

[Insert Figure 4.2 here]

Figure 4.2: Wesal Class Diagram

The diagram does not include separate classes for the administrator or the university supervisor because their main account information is represented by the `User` class and the `role` attribute. It also does not include a separate `Resume` class because the resume path is stored as part of the student profile. This keeps the class diagram focused on the main objects of the system and avoids unnecessary complexity.

4.3.2 Sequence Diagram

The sequence diagrams describe how the main users interact with the Wesal system during important operations. Each diagram presents the order of messages between the user, the web interface, the PHP application, and the database. The diagrams focus on the main steps rather than showing every internal PHP operation. This makes the system behavior easier to understand.

Sequence Diagram for Submitting an Application

This sequence diagram shows how a student applies for a training opportunity. First, the student views the published opportunities and selects one opportunity to see its details. After the student submits the application, the PHP application checks whether the student has already applied to the same opportunity. If no previous application exists, the system saves the application with the `Pending` status and displays a confirmation message. If a previous application exists, the system rejects the repeated application.

[Insert Figure 4.3 here]

Figure 4.3: Sequence Diagram for Submitting an Application

The process ensures that students can apply for different opportunities while preventing duplicate applications for the same opportunity. The application is stored in the database so that the student and the training organization can follow its status later.

Sequence Diagram for Reviewing an Application

This sequence diagram explains how a training organization reviews a student application. The organization first opens the list of applications received for its opportunities. The PHP application retrieves the relevant applications and displays the selected application details. The organization then chooses to accept or reject the application. If the application is rejected, its status is changed to `Rejected` ; if it is accepted, its status is changed to `Accepted` and a training record is created.

[Insert Figure 4.4 here]

Figure 4.4: Sequence Diagram for Reviewing an Application

The organization can only review applications related to its own training opportunities. This supports role-based access and prevents one organization from accessing the applications of another organization.

Sequence Diagram for Submitting and Reviewing a Report

This sequence diagram presents the process of submitting and reviewing a training report. The student opens an active training record, selects the report type, and uploads the report. The PHP application checks that the student has an active training record and then saves the report information with the `Submitted` status. The university supervisor can later

open the submitted report, add a comment, and update its status. The updated status and comment are saved in the database and can be viewed by the student.

[Insert Figure 4.5 here]

Figure 4.5: Sequence Diagram for Submitting and Reviewing a Report

The report process is intentionally simple because the project focuses on the main cooperative training workflow. The system supports the mid-training report and the final report without adding complex daily reporting or communication features.

4.3.3 Activity Diagram

The activity diagram shows the main workflow of the Wesal system from account registration to the completion of cooperative training. The student first registers and logs in, then completes the profile and uploads the resume and skills. After that, the student browses training opportunities and submits an application. The training organization reviews the application and either rejects it or accepts it and creates a training record.

After acceptance, the student starts the training and submits the required reports. The university supervisor reviews the reports and may request a revision when necessary. At the end of the training period, the training organization evaluates the student and the training is marked as completed. The diagram includes the main decision points without showing every small page or database operation.

[Insert Figure 4.6 here]

Figure 4.6: Wesal Cooperative Training Activity Diagram

The activity diagram also shows that a student can return to the opportunity search process after a rejected application. This supports the rule that a student can apply for several opportunities. However, after an accepted opportunity creates an active training record, the system should prevent the student from having another active training at the same time.

4.4 Database Design

The Wesal database is designed as a relational MySQL database. It stores the main information required to manage users, profiles, training opportunities, applications, training records, reports, and evaluations. The database uses primary keys to identify records and foreign keys to connect related records. The design also includes constraints to protect data consistency and prevent invalid operations.

The system uses one `users` table for all account types. The `role` field identifies whether the account belongs to a student, training organization, university supervisor, or administrator. Separate profile tables are used for students and training organizations because these users

have different information. The university supervisor and administrator do not require separate profile tables in the current project scope.

Main Database Entities

Entity	Purpose
`users`	Stores account, login, role, and account status information.
`student_profiles`	Stores the student's academic and personal information.
`organization_profiles`	Stores the training organization's information.
`skills`	Stores the skills used in student profiles.
`student_skills`	Connects students with their skills.
`training_opportunities`	Stores opportunities published by training organizations.
`applications`	Stores applications submitted by students.
`trainings`	Stores accepted training placements.
`reports`	Stores the reports submitted during training.
`evaluations`	Stores the evaluation submitted by the training organization.

Main Tables and Attributes

Users Table

The `users` table stores the common information for all system users. The `role` field is used to apply role-based permissions, and the `status` field can be used to activate or deactivate an account.

Attribute	Description
`user_id`	Primary key for the user account.
`full_name`	Full name of the user.
`email`	Unique email used for login.

password	Encrypted user password.
role	User role in the system.
status	Account status.
created_at	Account creation date.

Student Profiles Table

The student_profiles table stores the information required to identify and match a student with training opportunities. It includes the student's university number, major, academic level, contact information, location, and resume path.

Attribute	Description
student_id	Primary key for the student profile.
user_id	Foreign key connected to the users table.
university_id	Student university identification number.
major	Student academic major.
academic_level	Current academic level.
phone	Student contact number.
location	Student location.
resume_path	Path of the uploaded resume.

Organization Profiles Table

The organization_profiles table stores information about training organizations. It is connected to the user account that represents the organization.

Attribute	Description
organization_id	Primary key for the organization profile.
user_id	Foreign key connected to the users table.
organization_name	Name of the training organization.

description	Short description of the organization.
phone	Organization contact number.
location	Organization location.

Skills and Student Skills Tables

The skills table stores the available skill names. The student_skills table connects students with their skills because one student can have many skills and one skill can belong to many students.

Training Opportunities Table

The training_opportunities table stores the opportunities created by training organizations. It includes the opportunity details, required major, required skills, location, training dates, and publication status.

Applications Table

The applications table connects students with training opportunities. It stores the application status and the important dates. A unique constraint on student_id and opportunity_id prevents the same student from applying to the same opportunity more than once.

Trainings Table

The trainings table stores the training record created after an application is accepted. It includes the student, the opportunity, the training dates, and the training status.

Reports Table

The reports table stores the reports submitted during training. The system supports two simple report types: a mid-training report and a final report. The table also stores the review status and the supervisor's comment.

Evaluations Table

The evaluations table stores the evaluation submitted by the training organization. The evaluation includes a score, comments, and the evaluation date. The score can be represented on a scale from 1 to 5.

[Insert Figure 4.7 here]

Figure 4.7: Wesal Entity Relationship Diagram

Main Database Relationships

The `users` table has a one-to-zero-or-one relationship with `student_profiles` and `organization_profiles`. One organization can publish many training opportunities. One student can submit many applications, and one training opportunity can receive many applications. An accepted application can create one training record. One training record can contain several reports and can have one evaluation.

The relationship between students and skills is many-to-many and is implemented through the `student_skills` table. This structure allows each student to have several skills and allows the same skill to be used by many students. The database relationships support the recommendation function because the system can compare student skills and major with the requirements of training opportunities.

Database Constraints

The following constraints are used to maintain the reliability of the database:

1. Each user email must be unique.
2. Each application must belong to an existing student and training opportunity.
3. A student cannot apply to the same opportunity more than once.
4. A training record can only be created after an application is accepted.
5. A report can only be submitted by a student who has an accepted training record.
6. An organization can manage only its own training opportunities and applications.
7. Access to system operations is restricted according to the user's role.
8. The evaluation score must be within the selected score range.

4.5 User Interface Design

The user interface of Wesal is designed to be simple, clear, and responsive. Bootstrap is planned to support the layout on different screen sizes, while CSS is used for the project-specific visual style. JavaScript is used for simple interactions and form validation where needed. The interface will use common navigation elements so that users can understand the system without extensive instructions.

The system will provide different pages according to the user's role. Students will be able to access their dashboard, profile, resume, skills, training opportunities, applications, training records, reports, evaluations, and recommended opportunities. Training organizations will be able to access their organization profile, published opportunities, received applications, accepted students, and evaluation forms. University supervisors will be able to access

student information, application statuses, active training records, and submitted reports. Administrators will be able to manage user accounts, organizations, opportunities, and general system information.

The interface will display important statuses clearly. For example, application statuses such as `Pending`, `Accepted`, and `Rejected` will be shown using clear labels. Report statuses will also be displayed so that students and supervisors can understand whether a report has been submitted, reviewed, or approved. Forms will include clear field labels and validation messages to reduce incorrect data entry.

The detailed interface mockups and final screenshots will be added after the main pages are designed and implemented. They will be presented in this section or in Chapter 5 according to whether they represent the design proposal or the implemented system. The final interface design should remain consistent with the system roles, database structure, and workflows described in this chapter.

4.6 Summary

This chapter presented the design of the Wesal cooperative training management system. The architectural design divided the system into the web interface, PHP application, and MySQL database. The class diagram identified the main system objects and their relationships, while the sequence diagrams described the processes of submitting applications, reviewing applications, and submitting and reviewing reports. The activity diagram presented the complete cooperative training workflow from registration to training completion.

The database design defined the main tables, relationships, and constraints needed to store the system information. The user interface design described the main interface principles and the functions available to each user role. Together, these designs provide a clear structure for implementing Wesal using PHP, HTML, CSS, JavaScript, Bootstrap, MySQL, and XAMPP. The next chapter presents the implementation tools, the developed system, the testing cases, and the results.