Manage Project Homework Design Document

Design phase

Alejandro Onatra Caro

Mónica María Lozano Romero

December 22nd 2011

Table of contents

Package Diagram Table

Architecture Design Table

Component Design Table

Database Design Table

1 Introduction

1.1 Purpose

This document explains the general and specific architecture of the system that ultimately will be implemented for the MPH (Manage Project Homework) project included in the course: *Software Engineering part two,* at the Politecnico di Milano.

The document intends to describe the architectural decisions taken in the design process and justify them, also serving as an input for the next phase of the development process of the system.

1.2 Scope

This is the first iteration of the design process of the MPH system. Accordingly the scope of the architecture design is based on the analysis document presented in the analysis phase; as such, this iteration will not provide all the functionalities described as *functional requirements* in the analysis document.

The MPH system architecture in this iteration will be designed taking into account the implementation of the following general functions, organized according to four categories:

- **Manage projects**
  MPH will allow a professor, create and consult projects.

- **Manage project delivery specification**
  MPH will allow creating delivery definitions by the professor.

- **Manage project deliveries**
  MPH will allow students uploading, consulting, and updating deliveries. Users with the professor profile can download deliveries according to two criteria's.

- **Manage users**
  MPH will allow managing three types of users: administrator user, student and professor. Allowing each type of user, to consume some specific services of the system dedicated for it.

- **Manage project teams**
  MPH will manage the project teams, allowing them to deliver homework assignments to the involved project. For instance, the system will allow to the students create and subscribe to a project team, as well, it will let to the professors consult teams members.

1.3 Definition, acronyms, and abbreviations

The following acronyms will be used through the whole document:

- MPH: Manage Project Homework
- F: Functional requirement.
- NFR: Non-functional requirement.
- QA: Quality attribute.
- G: Goal.
- PM: Project manager
- DDM: Delivery specification manager
- DM: Delivery manager
- TM: Team manager
- UM: User manager
- JEE: Java platform, enterprise edition
- QoS: Quality of service
- AS: Application server
- EJB: Enterprise java bean
- JB: Java Bean
- POJO: Plain old java objects

The following definitions are relevant in all the sections of this document:

- *Delivery specification:* A description of a project delivery assignment
- *Delivery:* Uploaded file by a group for a specific delivery definition of a project.
- *Penalty:* A grade penalty for a late delivery done by a group.
- *Document type:* The document types are classes of documents predefined on the system that are assigned to delivery definitions.

1.4 References

The following references were used to specify this document:

- ProjectMPHdescription.pdf
- DesignArchitecturePartI.pdf
- DesignArchitecturePartII.pdf

1.5 Overview

This document specifies the architecture of the system in a general and specific way, using different levels of detail. Also describes the architectural decisions and justifies them. The design was developed in a top-down manner and so is the document containing the more general descriptions of the system at first and later discussing the specific details of each component.

The document is organized in the following sections:

1. **Introduction**

This section describes the purpose of the project and its general functional restrictions. Also introduces to the reader the set of important words and language used in this document and the general description of its structure.

2. **Design overview**
   Provide a general description of the software system including its functionality and matters related to the overall system and its design.

3. **Design considerations**
   This section describes the design assumptions and constrains of the system, specifies the requirements for optimal performance of the MPH and the interface design.

4. **System architecture**
   The general system architecture is specified in this section, describes the basic structure and interactions of the main subsystems of the MPH. Introducing a discussion of the main architectural decisions.

5. **Detail system design**
   Presents in more detail, trough different architectural views, the architecture of the system; specifying in with greater detail all the components of the system.

6. **Appendixes**
   Present a compilation of extra reference material for this document.

2 Design Overview

This section of the design document provides a general description of the design of the system and its process; includes the general design context, the general approach and describes the overall design.

2.1 Design context

The design context describes the basic limits for the system design, discussing the functional and technological context.

2.1.1 Functionalities

The functionalities that will be implemented and therefore the ones that will be included in the design phase will be presented in this section. The functionalities will be organized according to five relevant topics:

- Manage users.
- Manage projects.
- Manage team deliveries
- Manage teams
- Manage delivery specifications

The following functionalities were derived from the functional requirement found in the analysis document, the nonfunctional specification and in general the QoS details associated to the design will be discussed in section 3.3.

Manage projects

[F1] Create project
[F2] Consult project
[F3] Consult projects that belongs to a professor

Manage delivery specifications

[F4] Create project delivery specification
[F5] Consult project delivery specification

Manage deliveries

[F6] Consult deliveries specifications of a project
[F7] Download project deliveries by type
[F8] Download project deliveries by group
[F9] Assign grade to an uploaded delivery homework assignment
[F10] Upload a delivery homework assignment
[F11] Update a delivery homework assignment

[F12] Consult team deliveries
[F13] Calculate group final project grade

Manage users

[F14] Create Professor
[F15] Consult user profile
[F16] Register to the system
[F17] Login

Manage teams

[F18] Create a project team
[F19] Subscribe to a project team
[F20] Consult teams
[F21] Consult team

2.1.2 System technologies

The MPH system will use a 3-tier architecture, distributed approach; for each tier a type of technology is used:

Client tier

The client tier will use user interfaces coded in HTML 5.0, this is a markup language combine with the Java Server Faces technology, this is a server-side user interface component framework specially design for Java technology-based web applications.

Business tier

The language that will be used for the system implementation is *Java*, using specifically the *Java Platform, Enterprise Edition or JEE5*. This platform is based on the client-server paradigm and supports distributed, multi-tier system based on components. Specifically we use E*nterprise Java Beans* (EJB) version 3.0.

To support the platform we use as application server: *JBoss v5.1*, an open-source Java EE-based application server. This application server supports all the platform JEE5 features.

Persistence tier

To provide the data base management, *MySQL* v 5.5 a relational data base management system is used.

2.2 General design description

This section presents the basic general ideas of the architecture of the system in terms of the approach and the general models for encapsulation of the functionalities.

2.2.1 Design approach

The design approach is based in a multi-tier distributed system, specifically in a 3-tier one, where each is named as follows:

- *Client tier:* This tier of the system is in charge of interpreting the user actions and present to the user the information requested.
- *Business logic tier:* This tier process the data according to the requests from the client tier, using the data from the persistence tier.
- *Persistence tier:* This tier holds the information of the system data model, and is in charge of writing and sending the information requested by the business logic tier.

For designing the system a top-down approach is used. After the identification of the main three layers, the system is decomposed in components that capture subsets of related functionalities. For each component is specified (in detail) its responsibility on the architecture and its interactions with the other ones.

2.2.2 Overall design

In this section the general design schemas are presented specifying the basic relations between packages, user cases and users.

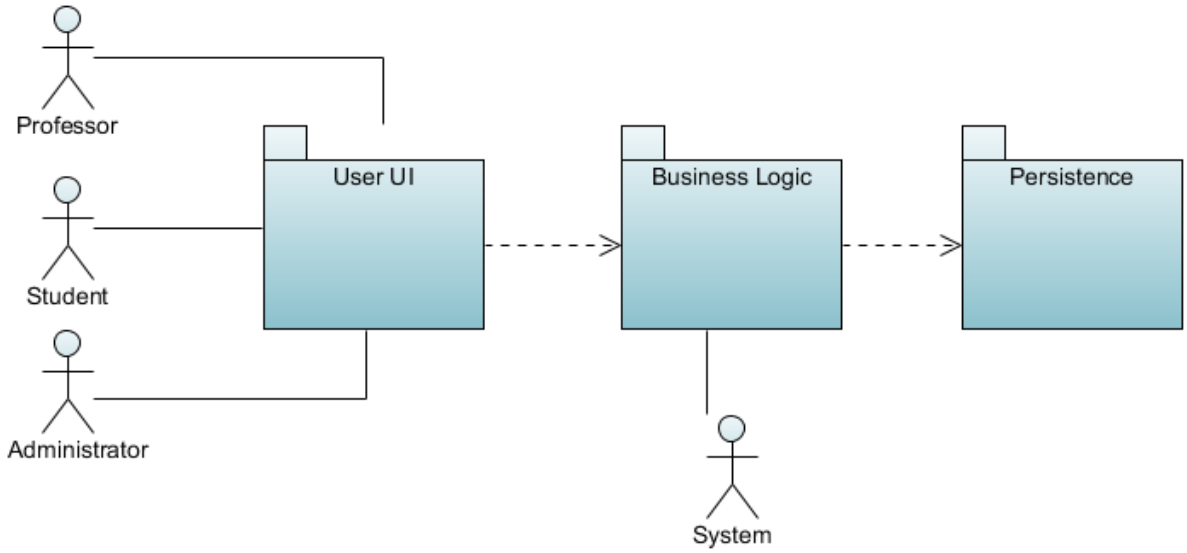
2.2.2.1 General package design

The system, as previously said, will be distributed in a three-layer fashion. Each layer contains a set of functionalities satisfying the correspondent requirements. Thus we find a mapping between the use cases and the system general basic package design.

In the diagram we can identify three packages:

- *User UI:* This package is in charge of interacting with the user: obtain the user requests, send this request to the *business logic* package, obtain the information needed from the latter one and display it to the user accordingly. In general the package contains the user interfaces.
- *Business logic:* This package is in charge of receiving the *User UI package* requests, processing them, accessing the *Persistence package* when needed and sending a response accordingly. In this package the business logic components are contained.
- *Persistence:* This package is in charge of managing the data request from the *Business logic package*.

The main users: Administrator, student and professor access directly the *User UI* package but cannot see the other packages, unlike the *system* user that have to access the some functionalities encapsulated by the Business logic, but neither of them can access directly the data.



Package Diagram 1: Basic package diagram

2.2.2.2 Detail Package design

Given the general package design according to the functionalities presented in the section 2.1.1 we can identify specific components within the packages as shown in the Package Diagram 2.

The inner packages are described as follows:

User UI

The set of sub packages composing this package is in charge in general of capturing the user actions and request the desire information according to them, with the respective Business Logic set of sub packages. Each package is also in charge of displaying the available options for its respective data category (i.e. projects, delivery specifications)

- *Team delivery pages:* This package will implement part of the following functionalities: **[F6] – [F15]**
- *Project pages:* This package will implement part of the following functionalities: **[F1] – [F3]**
- *Delivery Specification pages:* This package will implement part of the following functionalities: **[F4] and [F5]**
- *Team pages:* This package will implement part of the following functionalities: **[F22] – [F25]**
- *User pages:* This package will implement part of the following functionalities: **[F18] – [F21]**

Business logic

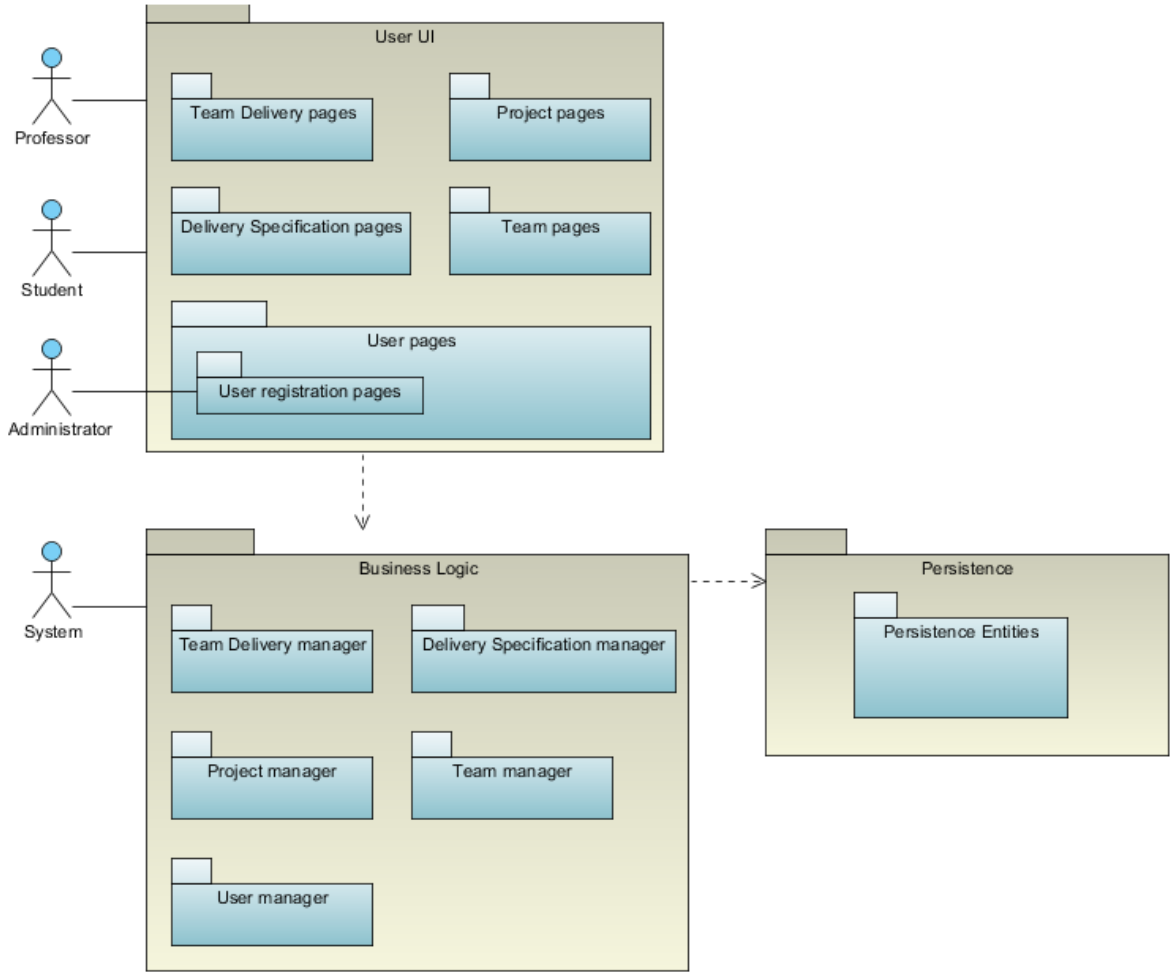
The set of sub packages composing this package is in charge in general of receiving the requests from the *User UI* package, processing them and send an answer back. The process can include accesses to the *Persistence package*.

- *Team delivery manager:* This package will implement part of the following functionalities: **[F6] – [F17]**
- *Project manager:* This package will implement part of the following functionalities: **[F1] – [F3]**
- *Delivery Specification manager:* This package will implement part of the following functionalities: **[F4] and [F5]**
- *Team manager:* This package will implement part of the following functionalities: **[F22] – [F25]**
- *User manager:* This package will implement part of the following functionalities: **[F18] – [F21]**

Persistence

The set of sub packages composing this package is in charge in general of contain the data structure of the system or the data base information. Receive the requests from the *Business logic package*, process it and send the desire data back.

- *Persistence entities:* This package will implement part of the following functionalities: **[F1] – [F25]**



Package Diagram 2: Specific package diagram

3 Design considerations

This section contains the design considerations taken into account in the MPH system design. Here is explained the assumptions and dependencies of the system, general constraints and the performance requirements satisfied by the current design.

3.1 Assumptions and dependencies

The following table resumes the assumptions done during the system design:

Assumption	Impact
The operating system where the software is installed has a java virtual machine.	The software only runs in operating systems that have java virtual machines available.
The file system where the team deliveries are saved is local.	The system won't have file transfer protocol implemented in this development phase.
The supported browsers will be Firefox, chrome or safari.	Is not assured the compatibility with internet explorer, given that it doesn't satisfy all the web standards.
Share team deliveries is no available	This functionality has to be done in the next version of the software.
A JEE application server runs in the server side	Without an application server the system can't be deployed.
There isn't an integration with an LDAP database	For using existing users by LDAP, this protocol must be implemented in future versions.

3.1.1 Dependencies and actions

Currently, the system doesn't have any kind of dependency (software, hardware, etc.).

Dependency	Action
<NA>	<NA>

3.2 General constraints

Constraints

Element	Requirements
Memory	At least 2 GB
Database server	My SQL
Network	Internet Access, HTTP protocol
Security	The system is controlled for each type of user SSL is not currently implemented
Hard disk space	At least 40 GB
Availability	Need to be tested and assured in future versions

3.3 Performance requirements

3.3.1 Standard compliance

Currently the software doesn't have standard compliances

3.3.2 Reliability

For assuring the reliability of the system, is necessary to do constant back up of the database and the file system of the server. Additionally, the integrity of the data saved in the database and in the file system must be assured.

3.3.3 Availability

For assuring the availability of the system, is used an application server. However for assuring a complete availability of the system is necessary to have redundancy in the application instances.

3.3.4 Security

The current design version doesn't include SSL in the user authentication, but it supports authorization according to the user profiles. Is recommended implementing SSL in future versions.

3.3.5 Maintainability

The architecture style and the component definition explained in the following chapters, assures low coupling and high cohesion. Therefore the system can be modified and improved easily.

3.3.6 Portability

The system is developed in java, which assure its portability to different operating systems and JEE application servers

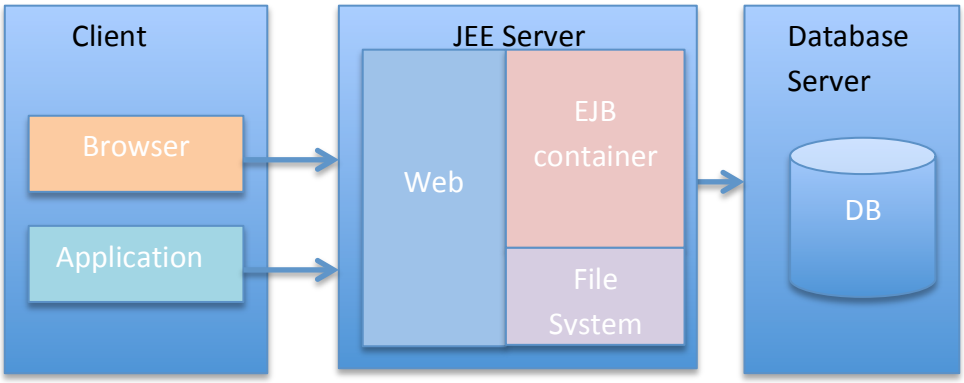
4 System Architecture

This section introduces the architecture of the system, such that the relevant components, topology and connectors are identified and explained. Furthermore, this section explains the use of the JEE technology in the MPH software.

4.1 General Architecture

The architectural design of the system is based on that the software to be developed is a web application and incorporates several Java EE technologies. It exposes the different component interfaces provided by enterprise beans. The enterprise beans use the Java persistence API to create and store the application data in the database. Additionally it contains different user interface pages such that they satisfy the software requirements.

The next diagram shows the client-server architecture of the system:



Architecture Design 1: Client-Server architecture

In this architecture scheme are satisfied the most important non-functional requirements (JEE technologies) such as concurrency, security, interoperability and exception control and management features. Additionally it allows that the database can be installed physically separate from the server in which the application server is installed.

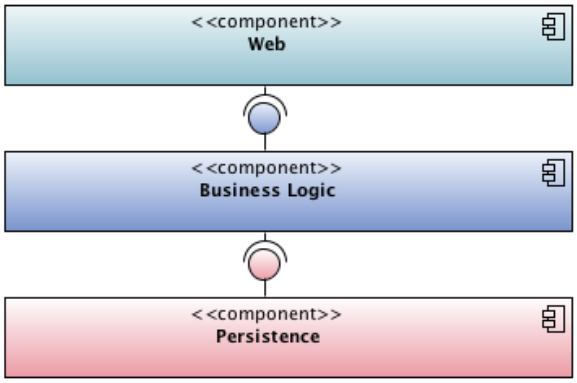
The MPH uses the following Java EE 5 platform features:

- Java persistence API entities
  - Java API for JavaBeans validation, annotations on the entities for verifying data.
- Enterprise beans:
  - Local, no-interface view session and singleton beans.
  - Java EE security constraints on the different interface business methods, based on user profile.
  - All the enterprise beans packaged within the EAR.
- Java Server Faces technology, using facelets for web front-end

- Templating
- Composite components
- AJAX-enabled facelets components.

4.2 Architecture and topology

The system architecture is based on a layer style where are clearly separated the presentation, business logic and the persistence of the system, as show in the following figure:



Architecture Design 2: Layer Architecture

For our design the previous architecture can be detailed in the specific components according to the software requirements, the figure Architecture Design 1 shows such components.

The components shown in the figure are the general components needed for implementing the MPH product, the next sections and chapters describe in detail design aspects for each of the components of each layer. Here we explain what components contain each one layer.

4.2.1 Web layer

This layer contains all the different kinds of web pages needed for interacting with each type of system user. Such pages are classified in for categories:

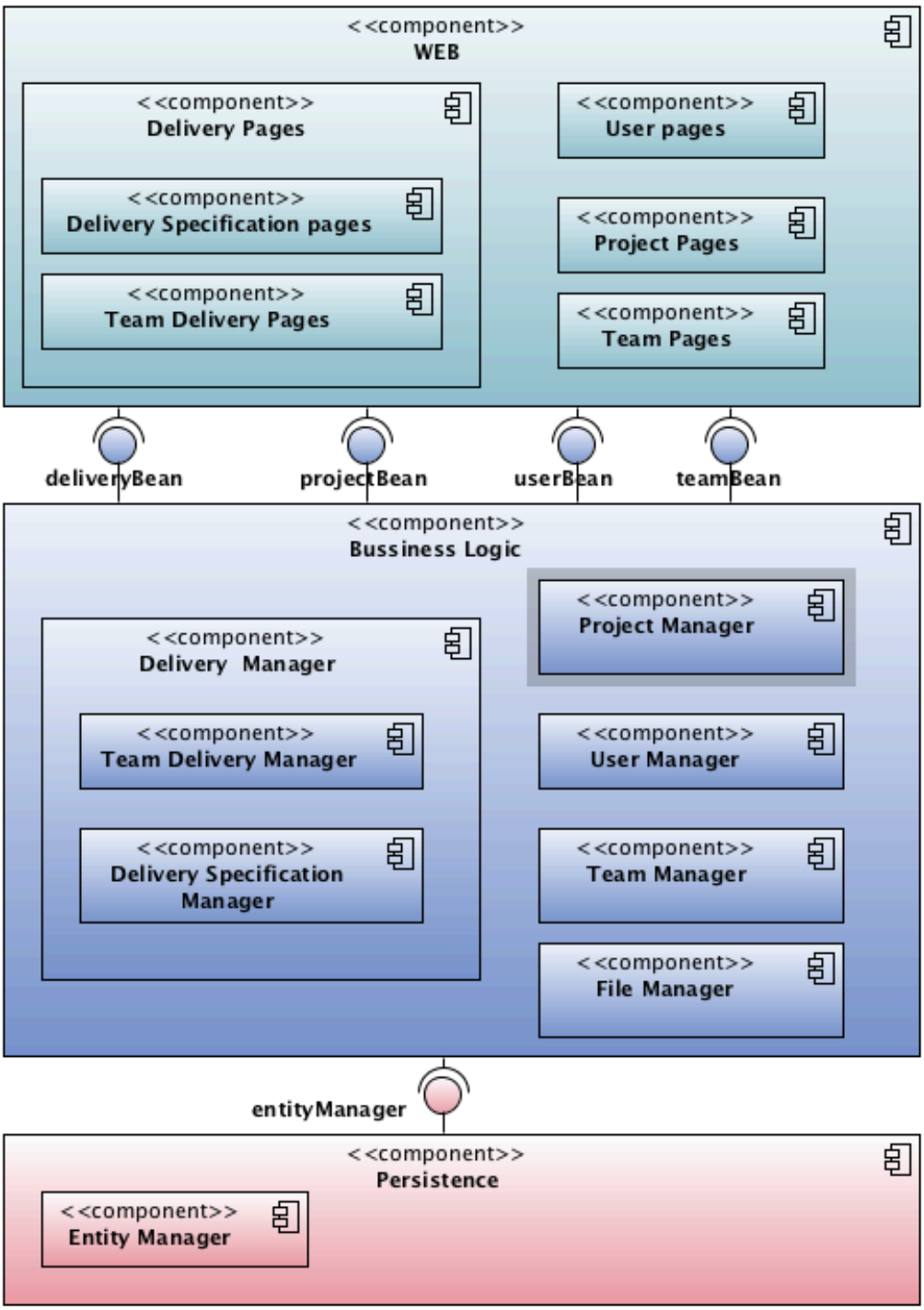
- **Delivery Pages**

  These pages are used for managing every use case related to the management of deliveries in the system. As shown in Architecture Design 3, it contains to different kind of pages:
  - Delivery specification pages
  - Team delivery pages

  In this way, the creation and management of delivery specification of a project is separated from the functionalities offered to the student-team deliveries.

- **User pages**

  These pages are used for all the use cases related to the user management such that login, registration, user creation, among others.



Architecture Design 3: MPH General Architecture

- **Project pages**
  These pages are used for all the use cases related to the management of projects, such that consultation and creation.

- **Team pages**

These pages are used for the use cases related to the management of teams, such that it creation, subscription and consultation.

4.2.2 Business Logic layer

This layer contains the components in charge of the management of the different functionalities of the system; such managers are classified in five categories:

- **Delivery Manager**
  This component is in charge of the management of all the functionalities related to the team deliveries and project delivery specification. This component contains two sub managers, such that the management of the project delivery specification and student-team deliveries is separated.

- **Project manager**
  This component is in charge of the management of all the functionalities related to the project management by a professor.

- **User manager**
  This component is in charge of the management of all the functionalities related to the user management. This includes the user authentication and authorization, professor user creation, student user registration, among others.

- **Team manager**
  This component is in charge of the management of all the functionalities related to the team management, such that the team creation and the student registration.

- **File System manager**
  This component is in charge of the management of the file system for the software, in this way it is in charge of the physical files for each team delivery.

This layer offers to the web layer four interfaces:

- Delivery Bean
- User Bean
- Project Bean
- Team Bean

The latter interfaces are based on method calling, such that the web layer (and so, all the web pages) invokes the respective interface based on the functionality required. The service provided for each one is specified in the next chapter.

4.2.3 Persistence Layer

This layer contains the components in charge of the entity management for the data managed in the system. It only contains the entities needed for saving the system data, the specific entities are specified in the next chapter.

This layer only offers one interface, which is the entity manager of the application server.

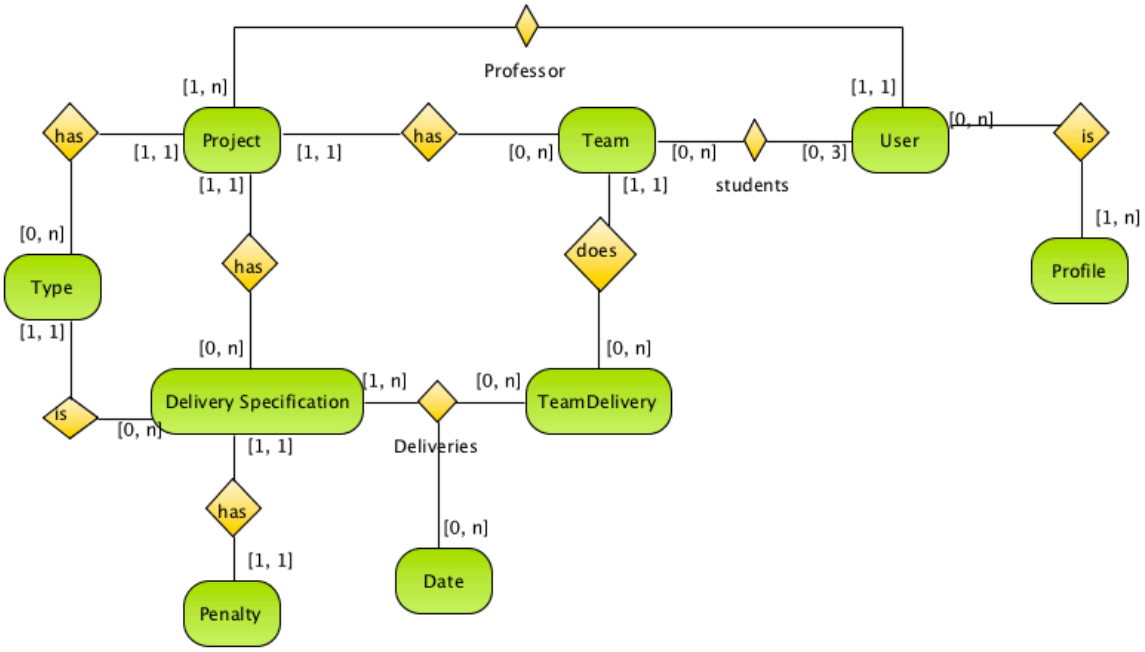
5 Detail System Design

This section explains in detail each component mentioned in the last chapter, specifying their responsibilities, methods, constraints, resources, among others. Additionally, in this section we define the database structure of the system in order to identify the JEE entities and they interactions.

5.1 Database Design

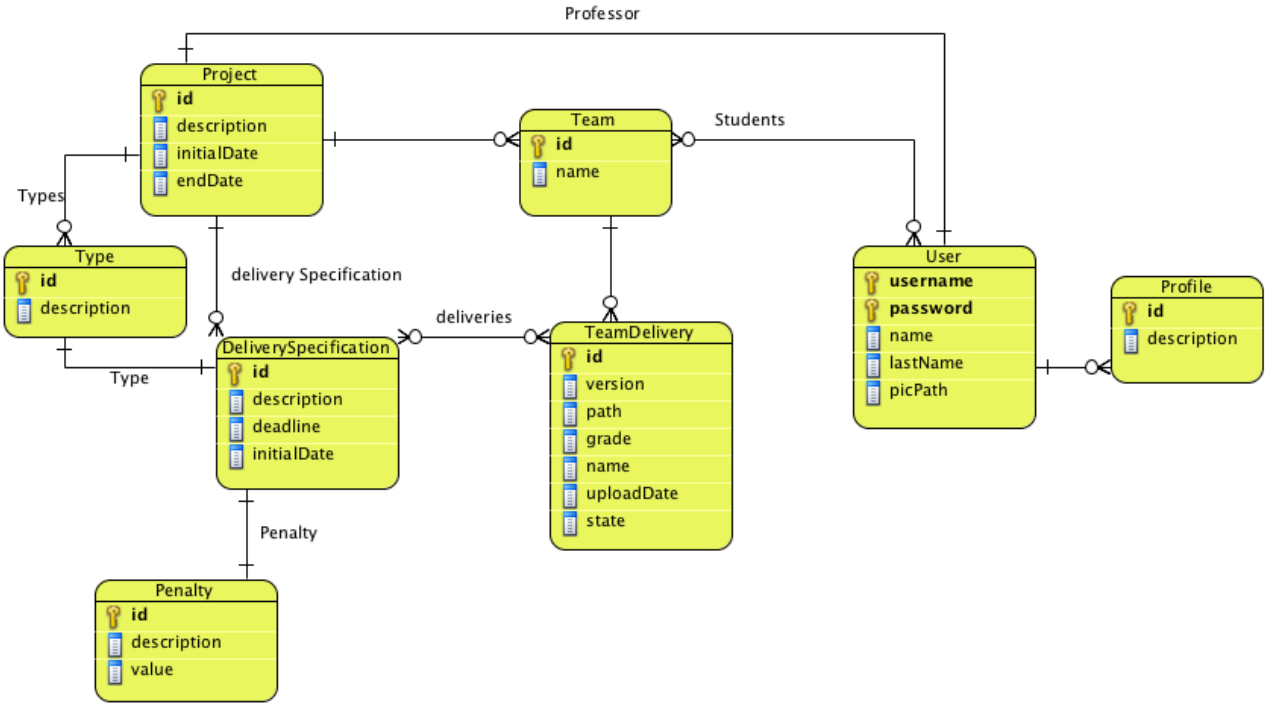
This subsection contains the database design (conceptual and logic design) , in which we are going to design the system entities.

5.1.1 Conceptual design



Database design 1 Conceptual Design

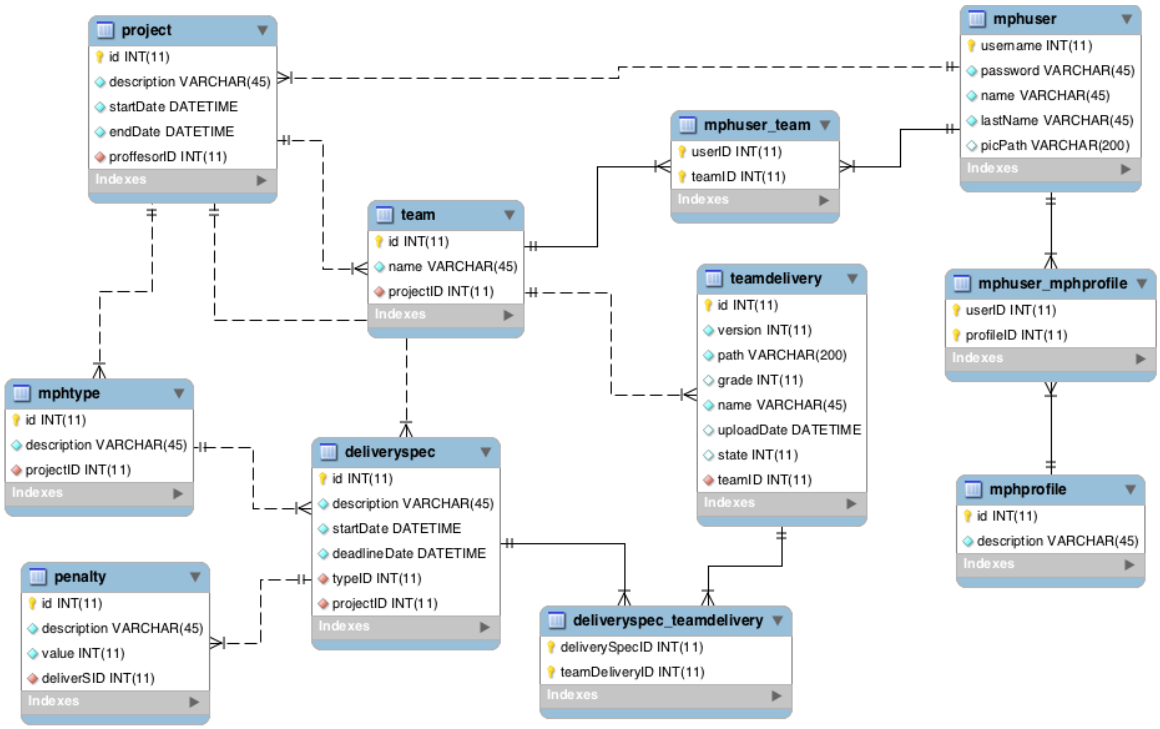
5.1.2 Logical design



Database design 2 Logical design

5.1.2.1 Tables

***Note**: These tables are automatically generated by JEE when the associated entities are created in the application server.*

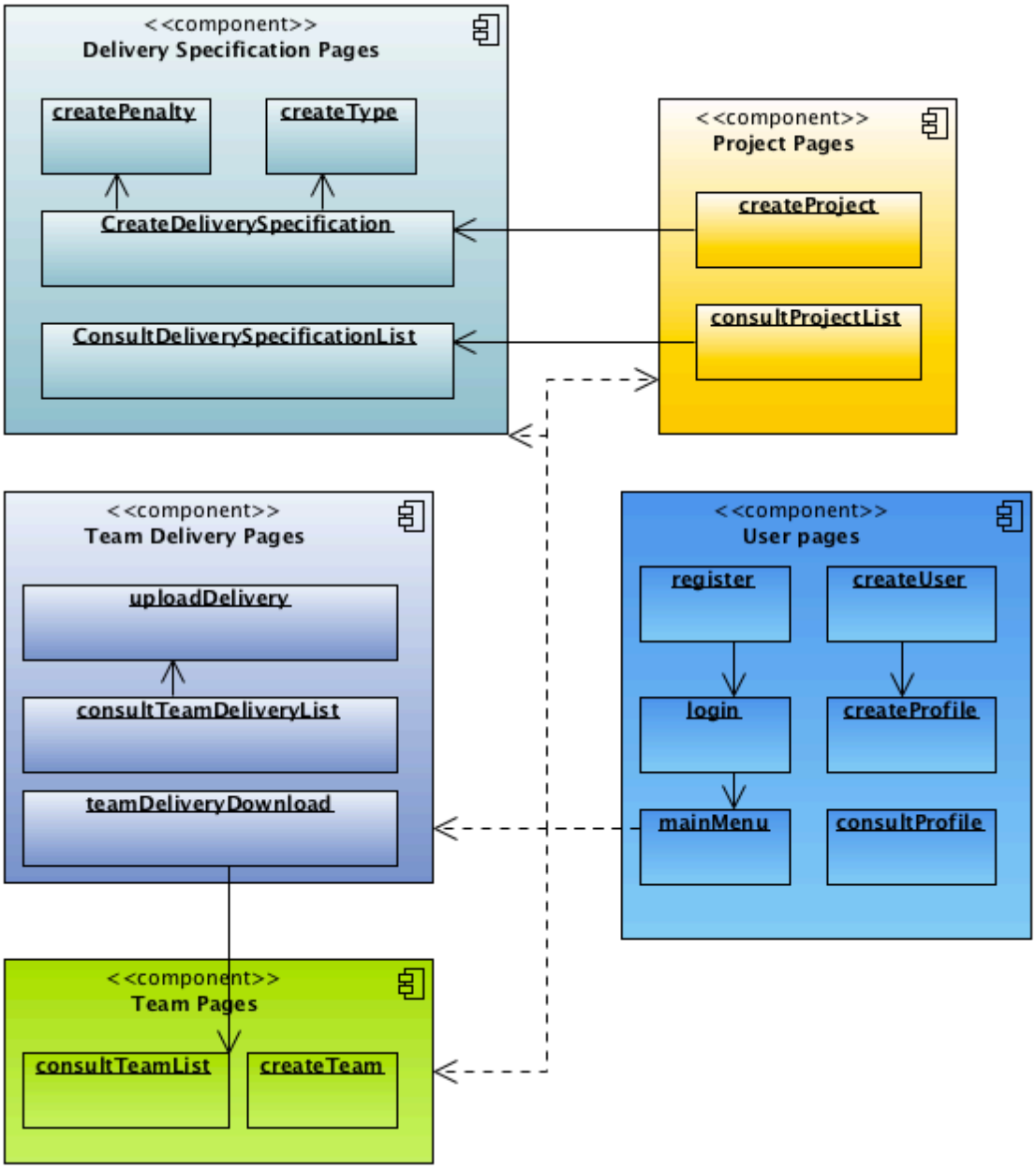


Database design 3 Database schema

5.2 Functional view

5.2.1 Web component

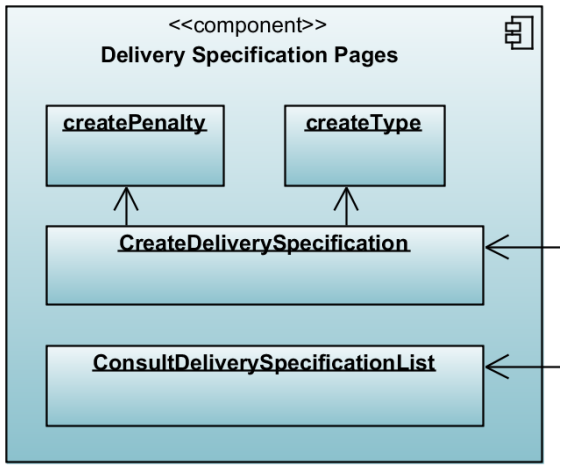
This section explains in detail each component of the web layer, the figure Component Design 7, shows how each component is compose and how they interact with each other.



Component Design 1: Web layer components

5.2.1.1 Delivery Specification Pages

The delivery specification pages are composed according to diagram Component Design 2:



Component Design 2: Delivery specification pages single component diagram

5.2.1.1.1 Create penalty

Create penalty	
Classification	Web page (CreatePenalty.xhtml)
Definition	User interface for the creation of a new penalty
Responsibilities	This component is responsible for: • Displaying to the user a dialog showing the form for creating a penalty • Capture the parameters given by the users as filled fields of the form to create a new *penalty.* • Send the parameters captured to the *deliveryBean*. • Display to the user if the creation of the new penalty was successful or not.
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • The parameters are valid for sending.
Composition	
Uses/ interactions	Used by the *createDeliverySpecification* page to create a new penalty while creating a new delivery specification.

5.2.1.1.2 Create type

Create type	
Classification	Web page (CreateType.xhtml)
Definition	User interface for defining a new *type* of *delivery specification.*
Responsibilities	This component is responsible for: • Displaying to the user a dialog showing the form for creating a new *delivery type* • Capture the parameters given by the users as filled fields of the form to create a new *delivery type.* • Send the parameters captured to the *deliveryBean*.
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • The parameters are valid for sending.
Composition	
Uses/ interactions	Used by the *createDeliverySpecification* page to create a new type while creating a new delivery specification.

5.2.1.1.3 Create delivery specification

Create delivery specification	
Classification	Web page(CreateDeliverySpecification.xhtml)
Definition	User interface for defining a delivery specification
Responsibilities	This component is responsible for: • Displaying to the user a dialog showing the options for creating a new *delivery specification.* • Given the options selected the component can choose to display to the user the *createType*, *createPenalty* or *createDelivery specification* interface. • Capture the parameters given by the users as filled fields of the form to create a new *delivery specification.* • Send the parameters captured to *deliveryBean.*

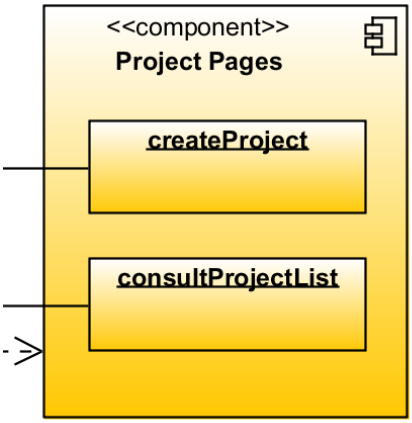
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • The parameters are valid for sending.
Composition	This component use: • Create type • Create penalty
Uses/ interactions	This interface uses the *createType and createPenalty* pages when the user, before creating a *delivery specification*, needs to create a new delivery type or a new penalty.

5.2.1.1.4 Consult delivery specification list

Consult delivery specification list	
Classification	Web Page(ConsultDeliverySpecificationList.xhtm)
Definition	User interface for displaying the list of available delivery specification for a given project to a user.
Responsibilities	This component is responsible for: • Display to the user the list of the available delivery specifications for a given project. • Display to the user the available options for managing each delivery specifications. • Obtain the list from the *deliveryBean.*
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • There is at least one project and at least one delivery specification available for the user.
Composition	
Uses/ interactions	To reach this page it's necessary to access first the *consultProjectList* page.

5.2.1.2 Project Pages

The project pages are composed according to diagram Component Design 3Component Design 3Component Design 2:



Component Design 3: Project pages single component view

5.2.1.2.1 Create project

Create project	
Classification	Web page(CreateProject.xhtml)
Definition	User interface for defining a project
Responsibilities	This component is responsible for: • Displaying to the user a dialog showing the options for creating a new *project.* • Capture the parameters given by the users as filled fields of the form to create a new *project.* • Send the parameters captured to *projectBean.* • Display to the user the available options for managing team options.
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • The parameters are valid for sending.
Composition	This component use: • Create delivery specification
Uses/ interactions	• When a user creates a new project and needs a new

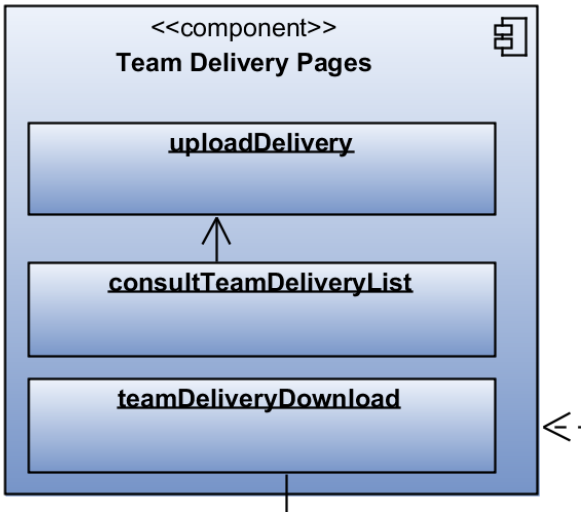
	delivery specification this web page calls *createDeliverySpecification*.

5.2.1.2.2 Consult project list

Consult project list	
Classification	Web Page(ConsultProjectList.xhtm)
Definition	User interface for displaying the list of available projects to a user.
Responsibilities	This component is responsible for: • Display to the user the list of the available projects to the user, calling the *projectBean.* • Display to the user the options to select a project for consulting the delivery specification list. • Display to the user the available options for managing group options.
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • There is at least one project and at least one delivery specification available for the user.
Composition	This component use: • Consult delivery specification list
Uses/ interactions	• When a user consults the available projects list and require further consult the delivery specifications of the project *consultDeliverySpecificationList* is used.

5.2.1.3 Team delivery pages

The team delivery pages are composed according to diagram Component Design 4Component Design 3Component Design 2:



Component Design 4: team delivery pages single component view

5.2.1.3.1 Upload delivery

Upload delivery	
Classification	Web page(UploadDelivery.xhtml)
Definition	User interface for uploading a project
Responsibilities	This component is responsible for: • Displaying to the user a dialog showing the form for uploading a *delivery.* • Capture the parameters given by the users as filled fields of the form and the selected delivery. • Send the parameters captured and the delivery file to *deliveryBean.*
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • The parameters are valid for sending. • The delivery file is valid for sending.
Composition	
Uses/ interactions	When a user selects the upload delivery option in the *consultTeamDeliveryList, the uploadDelivery* web page is displayed.

5.2.1.3.2 Consult team delivery list

Consult team delivery list	
Classification	Web Page(ConsultTeamDeliveryList.xhtm)
Definition	User interface for displaying the list of team deliveries to a user.
Responsibilities	This component is responsible for: • Display to the user the list of the available team deliveries to the user. • Display to the user the options to upload a delivery to the given delivery list corresponding to a delivery specification. • Obtain the list from the *deliveryBean.*
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • There is at least one project and one delivery specification and one delivery available for the user.
Composition	This component use: • Upload delivery
Uses/ interactions	When a user is located in the delivery list, an option for uploading a delivery is displayed, if selected, the *consultTeamDeliveryList* is used.

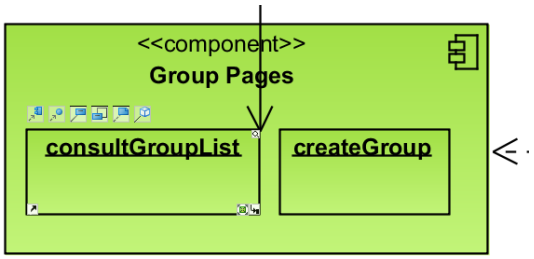
5.2.1.3.3 Team delivery download

Team delivery download	
Classification	Web page(TeamDeliveryDownload.xhtml)
Definition	User interface for downloading a team delivery
Responsibilities	This component is responsible for: • Displaying to the user a dialog showing the options for downloading a team delivery. • Display to the user the available options for managing group list and delivery specification type.

	• Capture the parameters given by the users as filled fields of the form to download a team delivery. • For obtaining the required types in case the user selects download by type: this page calls the *projectBean.* • For obtaining the required teams in case the user selects download by team: this page calls the *teamBean.* • Send the parameters captured to the *deliveryBean.* • Send to the user the list of team deliveries retrieved from the server.
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • The parameters are valid for sending.
Composition	This component use: • Consult team list
Uses/ interactions	If the user chooses download by team: The list of available teams is displayed using *consultTeamsList.*

5.2.1.4 Team pages

The team pages are composed according to diagram Component Design 5Component Design 3Component Design 2:



Component Design 5: Team pages single component view

5.2.1.4.1 Consult team list

Consult team list	
Classification	Web Page(ConsultTeamList.xhtm)
Definition	User interface for displaying the list of teams to a user.

Responsibilities	This component is responsible for: • Display to the user the list of the available teams, using the *teamBean.*
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • There is at least one team available for the user.
Composition	
Uses/ interactions	

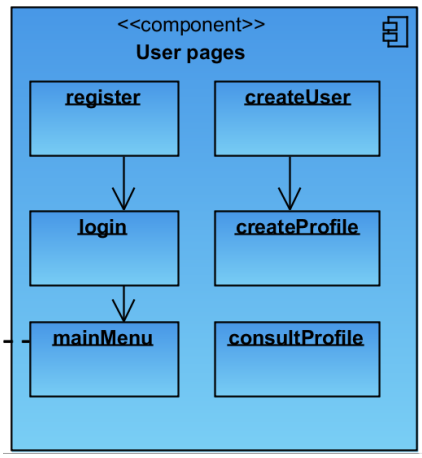
5.2.1.4.2 Create team

Create team	
Classification	Web page(CreateTeam.xhtml)
Definition	User interface for defining a team
Responsibilities	This component is responsible for: • Displaying to the user a dialog showing the options for creating a new *team.* • Capture the parameters given by the users as filled fields of the form to create a new *team.* • Send the parameters captured to *projectBean and userBean.* • Display to the user the available options for managing project options.
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • The parameters are valid for sending. • There is at least a project available for the user. • The team must be associated to a project after finishing the page confirm the success of the operation.
Composition	

Uses/ interactions	

5.2.1.5 User pages

The team pages are composed according to diagram Component Design 6Component Design 3Component Design 2:



Component Design 6: User pages single component view

5.2.1.5.1 Register

Register	
Classification	Web page(Register.xhtml)
Definition	User interface for registering a user to the system.
Responsibilities	This component is responsible for: • Displaying to the user a dialog showing the options for registering a new user. • Capture the parameters given by the users as filled fields of the form to register a new user. • Send the parameters captured to the *userBean.* • Confirm the success or not of the operation to the user.
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • The parameters are valid for sending.

Composition	This component use: • Login
Uses/ interactions	When the user finishes its registration, the *login* page is displayed.

5.2.1.5.2 Login

Login	
Classification	Web page(Login.xhtml)
Definition	User interface for downloading a team delivery
Responsibilities	This component is responsible for: • Displaying to the user a dialog showing the required information for the log in: *password* and *username*. • Capture the parameters given by the users as filled fields of the form confirm them with the information in the system. • Check the validity sending the information to the *userBean.* • If the user is valid display the *mainMenu* page, if not display an error message.
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • The parameters are valid for sending.
Composition	This component use: • Main menu
Uses/ interactions	When the user finishes its login process successfully, the *mainMenu* page is displayed.

5.2.1.5.3 Main menu

Main menu

Classification	Web page(MainMenu.xhtml)
Definition	User interface for displaying all the available actions for a user.
Responsibilities	This component is responsible for: • Displaying to the user a dialog showing all the main available actions related to: teams, deliveries, projects and user. • Redirect the user to the respective pages given its actions.
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active.
Composition	This component use: • Delivery specification pages • Project pages • Team delivery pages • Team pages
Uses/ interactions	• If the user chooses to go to delivery specification options, *mainMenu* use the corresponding page. • If the user chooses to go to project options, *mainMenu* use the corresponding page. • If the user chooses to go to team delivery options, *mainMenu* use the corresponding page. • If the user chooses to go to team options, *mainMenu* use the corresponding page.

5.2.1.5.4 Create user

Create user	
Classification	Web page(CreateUser.xhtml)
Definition	User interface for creating a new type of user.
Responsibilities	This component is responsible for: • Displaying to the user a dialog showing the options for

	creating a new type of user. • Capture the parameters given by the users as filled fields of the form to register a new user. • Send the parameters captured to the *userBean.* • Confirm the success or not of the operation to the user.
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • The parameters are valid for sending.
Composition	This component use: • Create profile
Uses/ interactions	When assigning a profile to the user, if doesn't exist, this page uses *createProfile.*

5.2.1.5.5 Create profile

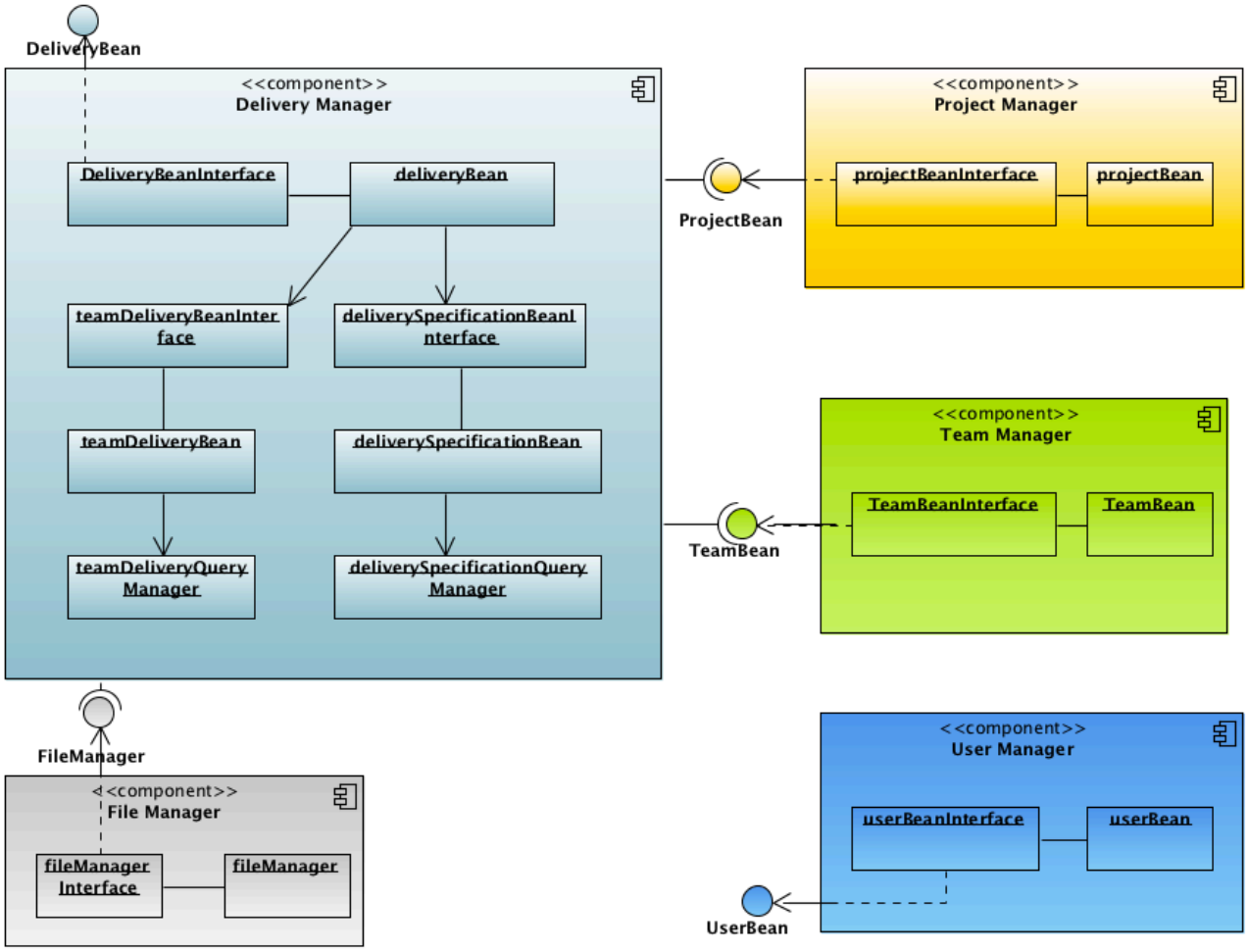
Create profile	
Classification	Web page(CreateProfile.xhtml)
Definition	User interface for creating a new profile.
Responsibilities	This component is responsible for: • Displaying to the user a dialog showing the options for creating a new profile. • Capture the parameters given by the users as filled fields of the form to create a new profile. • Send the parameters captured to the *userBean.* • Confirm the success or not of the operation to the user.
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active. • The parameters are valid for sending.
Composition	
Uses/ interactions	

5.2.1.5.6 Consult profile

Consult profile	
Classification	Web page(ConssultProfile.xhtml)
Definition	User interface for consulting the profile information by a user
Responsibilities	This component is responsible for: • Displaying to the user the personal information registered in the system. • Obtain the information through the *userBean.*
Constraints	• The load of the web page is completed. • The connection between the *persistence*, the *business logic* and the *web component* is active.
Composition	
Uses/ interactions	

5.2.2 Business logic components

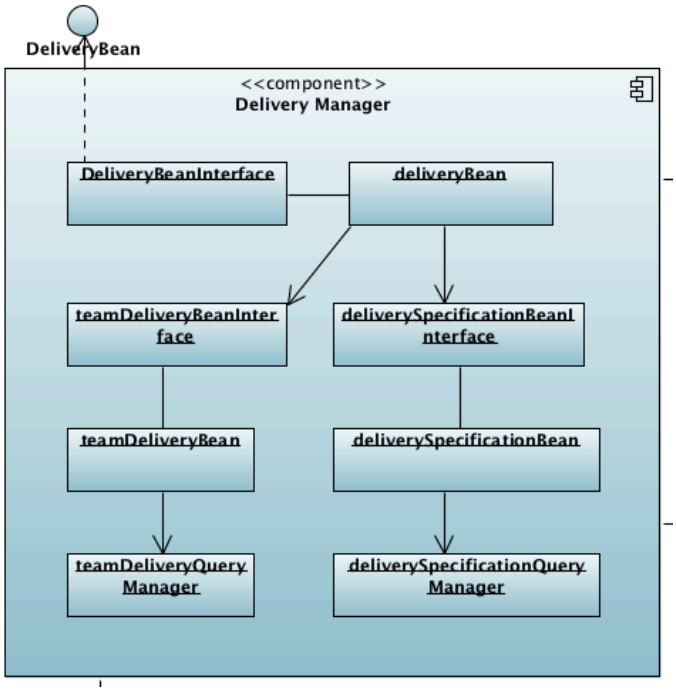
This section explains in detail each component of the business logic layer, the figure Component Design 7, shows how each component is compose and how they interact each other.



Component Design 7 Business logic component

In the following sections, each component and sub components are explained in detail.

5.2.2.1 Delivery Manager



Component Design 8 Delivery Manager

5.2.2.1.1 Delivery Bean

Delivery Bean (Interface/implementation)	
Classification	Bean: @Stateful
Definition	Is in charge in of all functionalities related to the deliveries management.
Responsibilities	This component does: • Create a delivery specification • Consult a delivery specification • Upload and update a team delivery • Download team delivery by type and team. • Search a team delivery • Search a delivery specification • Create delivery specification penalties • Create delivery specifications types
Constraints	This component is use by: • The delivery pages component in the web layer. • The project pages component in the web layer

Composition	This component use: • Team delivery bean • Delivery specification bean
Uses/ interactions	This component use the components mentioned in the latter literal according to the method invoked.
Resources	Uses the following entities: • Delivery Specification • Team Delivery • Penalty • Type
Processing	This component relay on the algorithms used in the components that it calls
Interface/Exports	This component publish the interface delivery Bean to the web layer and use the following interfaces: • Team Bean • Project Bean
Methods	
Create Delivery Specification	
Name	createDeliverySpecification
Parameters	projectID, name, description, initialDate, deadlineDate, penaltyID, typeID
Return Value	DeliverySpecification
Description	Creates a delivery specification
Data structure	DeliverySpecification
Precondition	The penalty and the type that is going to be associated already exist in the system
Validity Checks	The user is a professor
Post conditions	The delivery specification is created
Called by	CreatedeliverySpecification.xhtml

Calls	createDeliverySpecification(): DeliverySpecificationBean
Upload delivery	
Name	uploadDelivery
Parameters	deliverySpecID, teamID, file
Return Value	True/false
Description	Upload a delivery of a team to a delivery specification
Data structure	TeamDelivery
Precondition	The delivery specification already exists, and the team already exists
Validity Checks	The user is a student that belongs to the team
Post conditions	The file associated to the delivery is saved in the server. Previous file of the delivery are deleted
Called by	uploadDelivery.xhtml
Calls	uploadDelivery(): TeamDeliveryBean
Download deliveries by type	
Name	downloadDeliveriesByType
Parameters	projectID, typeID
Return Value	LinkedList<TeamDelivery)
Description	Search all the team deliveries of the project and creates a zip, for downloading
Data structure	LinkedList <TeamDelivery>
Precondition	The project and the type exists in the system
Validity Checks	The user is the professor of the project
Post conditions	A list of the deliveries is created, such that a zip can be done in other method (private)
Called by	TeamDeliveryDownload.xhtml
Calls	DownloadTeamDeliveryByType (): TeamDeliveryBean

Download deliveries by team	
Name	downloadDeliveriesByTeam
Parameters	projectID, teamID
Return Value	LinkedList<TeamDelivery)
Description	Search all the team deliveries of the project and creates a zip, for downloading.
Data structure	LinkedList <TeamDelivery>
Precondition	The project and the team exists in the system
Validity Checks	The user is the professor of the project
Postconditions	A list of the deliveries is created, such that a zip can be done in other method (private)
Called by	TeamDeliveryDownload.xhtml
Calls	DownloadTeamDeliveryByTeam (): TeamDeliveryBean
Create Penalty	
Name	createPenalty
Parameters	Name, description, deliverySpecificationID
Return Value	True/false
Description	Create a penalty
Data structure	Penalty
Precondition	
Validity Checks	The user is the professor of the project
Post conditions	The penalty is created
Called by	CreateDeliverySpecification.xhtml
Calls	createPenalty (): deliverySpecificationBean
Create type	
Name	createType

Parameters	Name, description
Return Value	True/false
Description	Create a delivery type
Data structure	Type
Precondition	
Validity Checks	The user is the professor of the project
Post conditions	The type is created
Called by	CreateDeliverySpecification.xhtml
Calls	createType (): DeliverySpecificationBean
Get all delivery specification	
Name	getAllDeliverySpecification
Parameters	projectID
Return Value	LinkedList<DeliverySpecification>
Description	Search all the delivery specification of a project
Data structure	LinkedList<DeliverySpecification>
Precondition	
Validity Checks	
Post conditions	All the delivery specifications of the project given is returned.
Called by	ConsultDeliverySpecificationList.xhtml
Calls	getAllDeliverySpecification (): DeliverySpecificationBean
Get delivery specification penalty	
Name	getDeliverySpecificationPenalty
Parameters	deliverySpecificationID
Return Value	Penalty

Description	Return the penalty of a delivery specification
Data structure	Penalty
Precondition	
Validity Checks	
Post conditions	All the delivery type of the project given is returned.
Called by	ConsultDeliverySpecification.xhtml
Calls	getAllTypes (): DeliverySpecificationBean
Get delivery specification type	
Name	getDeliverySpecificationType
Parameters	deliverySpecificationID
Return Value	Type
Description	Return the delivery specification type
Data structure	Type
Precondition	
Validity Checks	
Post conditions	The delivery type of the project given is returned.
Called by	ConsultDeliverySpecification.xhtml
Calls	getDeliverySpecificationType (): DeliverySpecificationBean
Get delivery specification project	
Name	getDeliverySpecificationProject
Parameters	deliverySpecificationID
Return Value	Project
Description	Return the project to which the delivery specification belongs
Data structure	Project

Precondition	
Validity Checks	
Post conditions	The project is returned.
Called by	ConsultDeliverySpecification.xhtml
Calls	getDeliverySpecificationProject (): DeliverySpecificationBean
Get delivery specification initial date	
Name	getDeliverySpecificationInitialDate
Parameters	deliverySpecificationID
Return Value	Date
Description	Return the start date of a delivery specification
Data structure	Date
Precondition	
Validity Checks	
Post conditions	The initial date is returned
Called by	ConsultDeliverySpecification.xhtml
Calls	getDeliverySpecificationInitialDate (): DeliverySpecificationBean
Get delivery specification deadline date	
Name	getDeliverySpecificationDeadlineDate
Parameters	deliverySpecificationID
Return Value	Date
Description	Return the deadline date of a delivery specification
Data structure	Date
Precondition	
Validity Checks	

Post conditions	The deadline date is returned
Called by	ConsultDeliverySpecification.xhtml
Calls	getDeliverySpecificationDeadlineDate (): DeliverySpecificationBean
Get all team deliveries	
Name	getAllTeamDeliveries
Parameters	teamID
Return Value	LinkedList<TeamDelivery>
Description	Return all the deliveries done by a student team
Data structure	LinkedList<TeamDelivery>
Precondition	
Validity Checks	The user is the student
Post conditions	The deadline date is returned
Called by	TeamDeliveryList.xhtml
Calls	getAllTeamDeliveries (): TeamDeliveryBean
Get the delivery specification of a team delivery	
Name	getDeliverySpecificationForTeamDelivery
Parameters	teamDeliveryID
Return Value	DeliverySpecification
Description	Return the delivery specification of a team delivery
Data structure	DeliverySpecification
Precondition	
Validity Checks	The user is the student
Post conditions	
Called by	TeamDeliveryList.xhtml

Calls	getDeliverySpecificationForTeamDelivery (): TeamDeliveryBean
Grade team delivery	
Name	gradeTeamDelivery
Parameters	teamDeliveryID, grade
Return Value	True/false
Description	Grade the team delivery
Data structure	
Precondition	
Validity Checks	The user is a professor
Post conditions	
Called by	ConsultTeamDelivery.xhtml
Calls	gradeTeamDelivery (): TeamDeliveryBean
Calculate team deliveries final grade	
Name	calculateTeamFinalGradeForProject
Parameters	teamID, projectID
Return Value	True/false
Description	Calculates the final grade of all the deliveries of a team
Data structure	
Precondition	
Validity Checks	The user is a professor (next version should do this automatically based on the final date of a project)
Post conditions	
Called by	ConsultTeamDelivery.xhtml
Calls	gradeTeamDelivery (): TeamDeliveryBean

5.2.2.1.2 Team Delivery Bean

Team Delivery Bean (Interface/implementation)	
Classification	Bean: @stateless
Definition	Is in charge in of all functionalities related to the team deliveries management.
Responsibilities	This component does: • Upload and update a team delivery • Search a team delivery • Download team deliveries by team and by group
Constraints	This component is use by: • The delivery bean.
Composition	This component use: • Team delivery query manager • File Manager
Uses/ interactions	This component use the components mentioned in the latter literal according to the method invoked.
Resources	This component uses the following entities: • Team Delivery • Team • Delivery specification
Processing	This bean do all the validation necessary for managing the team deliveries
Interface/Exports	
Methods	
Upload delivery	
Name	uploadDelivery
Parameters	deliverySpecID, teamID, file
Return Value	True/false
Description	Upload a delivery of a team to a delivery specification

Data structure	TeamDelivery
Precondition	The delivery specification already exists, and the team already exists
Validity Checks	The user is a student that belongs to the team
Post conditions	The file associated to the delivery is saved in the server. Previous file of the delivery are deleted
Called by	DeliveryBean
Calls	Save():FileManager
Download deliveries by type	
Name	downloadDeliveriesByType
Parameters	projectID, typeID
Return Value	LinkedList<TeamDelivery)
Description	Search all the team deliveries of the project and creates a zip, for downloading
Data structure	LinkedList <TeamDelivery>
Precondition	The project and the type exists in the system
Validity Checks	The user is the professor of the project
Post conditions	A list of the deliveries is created, such that a zip can be done in other method (private)
Called by	DeliveryBean
Calls	toZip():FileManager
Download deliveries by team	
Name	downloadDeliveriesByTeam
Parameters	projectID, teamID
Return Value	LinkedList<TeamDelivery)
Description	Search all the team deliveries of the project and creates a zip, for downloading.
Data structure	LinkedList <TeamDelivery>

Precondition	The project and the team exists in the system
Validity Checks	The user is the professor of the project
Postconditions	A list of the deliveries is created, such that a zip can be done in other method (private)
Called by	DeliveryBean
Calls	toZip():FileManager
Get all team deliveries	
Name	getAllTeamDeliveries
Parameters	teamID
Return Value	LinkedList<TeamDelivery>
Description	Return all the deliveries done by a student team
Data structure	LinkedList<TeamDelivery>
Precondition	
Validity Checks	The user is the student
Post conditions	The deadline date is returned
Called by	DeliveryBean
Calls	
Get the delivery specification of a team delivery	
Name	getDeliverySpecificationForTeamDelivery
Parameters	teamDeliveryID
Return Value	DeliverySpecification
Description	Return the delivery specification of a team delivery
Data structure	DeliverySpecification
Precondition	
Validity Checks	The user is the student

Post conditions	
Called by	DeliveryBean
Calls	
Grade team delivery	
Name	gradeTeamDelivery
Parameters	teamDeliveryID, grade
Return Value	True/false
Description	Grade the team delivery
Data structure	
Precondition	
Validity Checks	The user is a professor
Post conditions	
Called by	ConsultTeamDelivery.xhtml
Calls	gradeTeamDelivery (): TeamDeliveryBean
Calculate final grade of a team	
Name	calculateFinalGrade
Parameters	teamDeliveryID
Return Value	Double
Description	Calculates de final grade of a team
Data structure	Double
Precondition	
Validity Checks	Is called by the container
Post conditions	
Called by	ConsultTeamList.xhtml

Calls	

5.2.2.1.3 Delivery Specification Bean

Delivery Specification bean (Interface/implementation)	
Classification	Bean: @stateless
Definition	Is in charge in of all functionalities related to the deliveries specification management.
Responsibilities	This component does: • Create a delivery specification • Consult a delivery specification • Search a delivery specification • Create delivery specification penalties • Create delivery specifications types
Constraints	This component is use by: • Delivery Bean
Composition	This component use: • Delivery specification query manager
Uses/ interactions	This component use the components mentioned in the latter literal according to the method invoked.
Resources	Uses the following entities: • Delivery Specification • Team Delivery • Penalty • Type
Processing	This component validates and executes all the steps needed to accomplish each of the delivery specification's functionalities.
Interface/Exports	
Methods	

Create Delivery Specification	
Name	createDeliverySpecification
Parameters	projectID, name, description, initialDate, deadlineDate, penaltyID, typeID
Return Value	DeliverySpecification
Description	Creates a delivery specification
Data structure	DeliverySpecification
Precondition	The penalty and the type that is going to be associated already exist in the system
Validity Checks	The user is a professor
Post conditions	The delivery specification is created
Called by	DeliveryBean
Calls	
Create Penalty	
Name	createPenalty
Parameters	Name, description, deliverySpecificationID
Return Value	True/false
Description	Create a penalty
Data structure	Penalty
Precondition	
Validity Checks	The user is the professor of the project
Post conditions	The penalty is created
Called by	DeliveryBean
Calls	
Create type	
Name	createType

Parameters	Name, description
Return Value	True/false
Description	Create a delivery type
Data structure	Type
Precondition	
Validity Checks	The user is the professor of the project
Post conditions	The type is created
Called by	DeliveryBean
Calls	
Get all delivery specification	
Name	getAllDeliverySpecification
Parameters	projectID
Return Value	LinkedList<DeliverySpecification>
Description	Search all the delivery specification of a project
Data structure	LinkedList<DeliverySpecification>
Precondition	
Validity Checks	
Post conditions	All the delivery specifications of the project given is returned.
Called by	DeliveryBean
Calls	
Get delivery specification penalty	
Name	getDeliverySpecificationPenalty
Parameters	deliverySpecificationID
Return Value	Penalty

Description	Return the penalty of a delivery specification
Data structure	Penalty
Precondition	
Validity Checks	
Post conditions	All the delivery type of the project given is returned.
Called by	DeliveryBean
Calls	
Get delivery specification type	
Name	getDeliverySpecificationType
Parameters	deliverySpecificationID
Return Value	Type
Description	Return the delivery specification type
Data structure	Type
Precondition	
Validity Checks	
Post conditions	The delivery type of the project given is returned.
Called by	DeliveryBean
Calls	
Get delivery specification project	
Name	getDeliverySpecificationProject
Parameters	deliverySpecificationID
Return Value	Project
Description	Return the project to which the delivery specification belongs
Data structure	Project

Precondition	
Validity Checks	
Post conditions	The project is returned.
Called by	DeliveryBean
Calls	
Get initial date	
Name	getDeliverySpecificationInitialDate
Parameters	deliverySpecificationID
Return Value	Date
Description	Return the start date of a delivery specification
Data structure	Date
Precondition	
Validity Checks	
Post conditions	The initial date is returned
Called by	DeliveryBean
Calls	
Get deadline date	
Name	getDeliverySpecificationDeadlineDate
Parameters	deliverySpecificationID
Return Value	Date
Description	Return the deadline date of a delivery specification
Data structure	Date
Precondition	
Validity Checks	

Post conditions	The deadline date is returned
Called by	DeliveryBean
Calls	

5.2.2.1.4 Team Delivery Query Manager

Team delivery query manager	
Classification	POJO
Definition	Is in charge of executing all the EJB queries needed for the component team delivery bean.
Responsibilities	This component does: • Search a team delivery • Search a delivery specification of a team delivery ***Note**: More queries can appear in the development phase*
Constraints	This component is use by: • Team delivery Bean
Composition	
Uses/ interactions	This component executes EJB queries
Resources	Uses the following entities: • Delivery Specification • Team Delivery
Processing	This component executes EJB queries
Interface/Exports	
Methods	
Search team deliveries	
Name	getAllTeamDeliveries

Parameters	teamID
Return Value	LinkedList<TeamDelivery>
Description	Search the team deliveries given a team ID
Data structure	LinkedList<TeamDelivery>
Precondition	
Validity Checks	
Post conditions	The team deliveries is returned
Called by	TeamDeliveryBean
Calls	
Search team deliveries to a delivery specification	
Name	getAllTeamDeliveriesForDeliverySpecification
Parameters	teamID, deliverySpecificationID
Return Value	LinkedList<TeamDelivery>
Description	Search the team deliveries given a team ID and the delivery specification
Data structure	LinkedList<TeamDelivery>
Precondition	
Validity Checks	
Post conditions	The team deliveries is returned
Called by	TeamDeliveryBean
Calls	

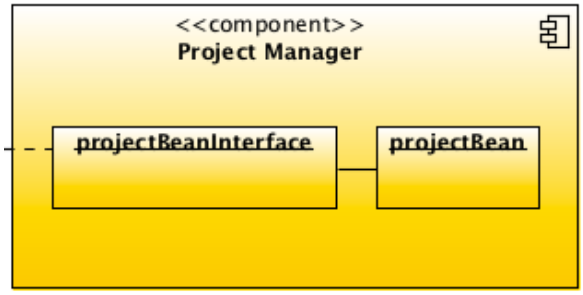
5.2.2.1.5 Delivery specification query manager

Delivery specification query manager	
Classification	POJO
Definition	Is in charge of executing all the EJB queries needed for the component delivery

	specification bean.
Responsibilities	This component does: • Search a delivery specifications with team delivery ***Note**: More queries can appear in the development phase*
Constraints	This component is use by: • Delivery specification Bean
Composition	
Uses/ interactions	This component executes EJB queries
Resources	Uses the following entities: • Delivery Specification • Team Delivery
Processing	This component executes EJB queries
Interface/Exports	
Methods	
Search delivery specification with team deliveries	
Name	searchTeamDeliveriesForDeliverySpecification
Parameters	deliverySpecificationID
Return Value	LinkedList<TeamDelivery>
Description	Search the team deliveries given a deliverySpecificationID
Data structure	LinkedList<TeamDelivery>
Precondition	
Validity Checks	
Post conditions	The team deliveries is returned
Called by	DeliverySpecificationBean

Calls	

5.2.2.2 Project Manager



Component Design 9 Project Manager

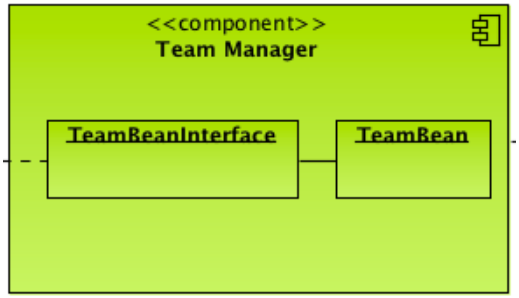
5.2.2.2.1 Project Bean

Project Bean (Interface/Implementation)	
Classification	Bean: @Stateless
Definition	Is in charge of all the functionalities related to the project management
Responsibilities	This component does: • Create a project • Consult the projects created by a professor
Constraints	This component is use by: • Create project web page.
Composition	
Uses/ interactions	This bean uses the entity manager for the project entity
Resources	Uses the following entities: • Project
Processing	This bean uses the entity manager for the project entity
Interface/Exports	This component exposes an interface for creating and consulting projects created by the system professors.
Methods	

Create project	
Name	createProject
Parameters	description
Return Value	True/false
Description	Creates a project
Data structure	Project
Precondition	The user is a professor
Validity Checks	
Post conditions	The project is created
Called by	CreateProject.xhtml
Calls	
Consult projects	
Name	consultProjects
Parameters	username
Return Value	LinkedList<Project>
Description	Return all the projects created by a professor
Data structure	LinkedList<Project>
Precondition	The user is a professor
Validity Checks	
Post conditions	All projects are searched
Called by	ConsultProjectList.xhtml
Calls	
Get all project types	
Name	getAllTypes

Parameters	projectID
Return Value	LinkedList<Type>
Description	Search all the delivery specification types in of project
Data structure	LinkedList<Type>
Precondition	
Validity Checks	The user is the professor of the project
Post conditions	All the delivery type of the project given is returned.
Called by	CreateDeliverySpecification.xhtml
Calls	getAllTypes (): DeliverySpecificationBean

5.2.2.3 Team Manager



Component Design 10 Team Manager

5.2.2.3.1 Team Bean

Team Bean (Interface/Implementation)	
Classification	Bean:@Stateless
Definition	Is in charge of all the functionalities related to the team management
Responsibilities	This component does: • Create a team • Consult the student teams • Subscribes an student

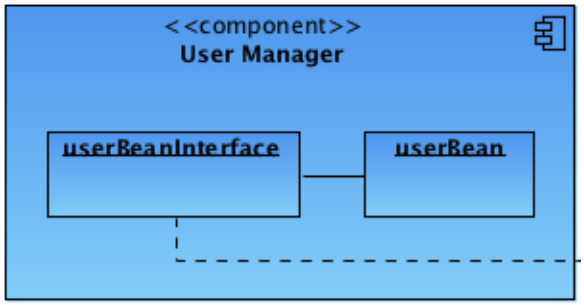
Constraints	This component is use by: • Create team web page. • Delivery Bean • Consult Team List web page
Composition	
Uses/ interactions	This bean uses the entity manager for the team entity
Resources	Uses the following entities: • Team
Processing	This bean uses the entity manager for the team entity
Interface/Exports	This component exposes an interface for creating, consulting and subscribing to teams.
Methods	
Create team	
Name	createTeam
Parameters	Name, studentID
Return Value	True/false
Description	Creates a team, with the student that is creating it
Data structure	Team
Precondition	The user is a student
Validity Checks	
Post conditions	The team is created
Called by	CreateTeam.xhtml
Calls	
Consult teams	
Name	consultTeam

Parameters	userID
Return Value	LinkedList<Team>
Description	Return all the teams to which the student belongs to
Data structure	LinkedList<Team>
Precondition	The user is a student
Validity Checks	
Post conditions	All teams to which the student belongs are searched
Called by	ConsultTeamList.xhtml
Calls	
Subscribe to a team	
Name	subscribeToTeam
Parameters	userID, teamID
Return Value	True/false
Description	The student is subscribe to the team selected
Data structure	Team
Precondition	The user is a student
Validity Checks	
Post conditions	The student belongs to the team
Called by	ConsultTeamList.xhtml
Calls	
Get all teams of a project	
Name	getAllTeamsForProject
Parameters	projectID
Return Value	LinkedList<Team>

Description	Returns all the teams in a project
Data structure	LinkedList<Team>
Precondition	
Validity Checks	
Post conditions	The teams belongs to the given project
Called by	ConsultTeamList.xhtml
Calls	
Get all the students of a team	
Name	getAllStudentsOfTeam
Parameters	teamID
Return Value	LinkedList<User>
Description	Returns all the students of the given team
Data structure	LinkedList<User>
Precondition	
Validity Checks	
Post conditions	The returned user has student profile
Called by	ConsultTeamList.xhtml
Calls	
Get all the team of a given student	
Name	getAllTeamsOfStudent
Parameters	sUsername
Return Value	LinkedList<Team>
Description	Returns all the teams to which the student belongs to in the different projects in which he/her participate
Data structure	LinkedList<Team>

Precondition	
Validity Checks	
Post conditions	The returned user has student profile
Called by	ConsultTeamList.xhtml
Calls	

5.2.2.4 User Manager



Component Design 11 User Manager

5.2.2.4.1 User bean

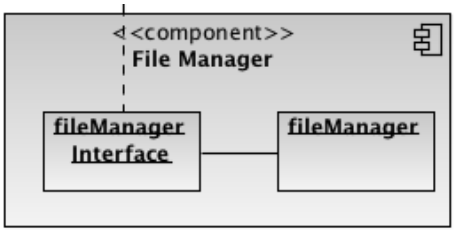
User Bean (Interface/Implementation)	
Classification	Bean:@Stateful
Definition	Is in charge of all the functionalities related to the user management
Responsibilities	This component does: • Create a professor • Login a user • Register an student • Create and consult a profile
Constraints	This component is use by: • Login web page • Register web page • Create user web page

	• Create a profile web page • Consult a profile web page
Composition	
Uses/ interactions	This bean uses the entity manager for the user and profile entity
Resources	Uses the following entities: • User • Profile
Processing	This bean uses the entity manager for the user and profile entity
Interface/Exports	This component exposes an interface for creating and consulting the user and profile information.
Methods	
Create Professor	
Name	createProfessorUser
Parameters	Username, password, name, lastName, picPath
Return Value	True/false
Description	Creates a professor
Data structure	User
Precondition	The user is the administrator
Validity Checks	
Post conditions	The user is created
Called by	CreateUser.xhtml
Calls	
Consult profile	
Name	consultProfile
Parameters	Username

Return Value	Profile
Description	Return the user profile
Data structure	Profile
Precondition	
Validity Checks	
Post conditions	The profile is returned
Called by	ConsultProfile.xhtml
Calls	
Login	
Name	login
Parameters	Username, password
Return Value	True/false
Description	Login a user
Data structure	User
Precondition	
Validity Checks	
Post conditions	The user related to the username/password
Called by	login.xhtml
Calls	
Register Student	
Name	registerStudent
Parameters	Username, password, name, lastName, picPath
Return Value	True/false
Description	Registers a user

Data structure	User
Precondition	The user must be a student
Validity Checks	
Post conditions	An user is created
Called by	registerStudent.xhtml
Calls	
Create profile	
Name	createProfile
Parameters	description
Return Value	True/false
Description	Creates a profile
Data structure	Profile
Precondition	The profile does not exists
Validity Checks	
Post conditions	The profile is created
Called by	createProfile.xhtml
Calls	

5.2.2.5 File Manager



Component Design 12 File Manager

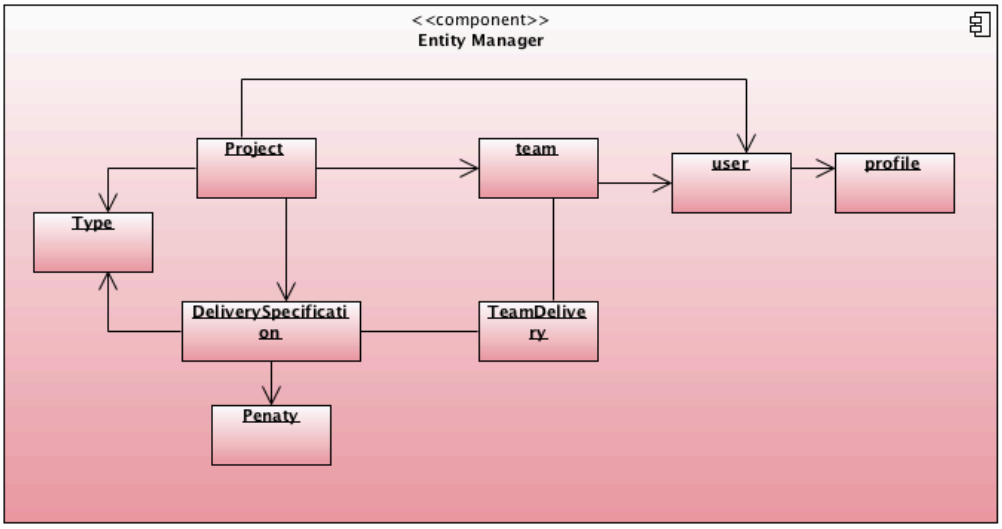
5.2.2.5.1 File Manager

File Manager (Interface/Implementation)

Classification	Component
Definition	Is in charge of the physical management of the team deliveries
Responsibilities	This component does: • Save a file in the server • Zip a set of files
Constraints	This component is use by: • Team delivery Bean
Composition	
Uses/ interactions	This component uses the file classes of java.
Resources	Uses the following entities: • Team Delivery
Processing	This component uses the file classes of java. Additionally it manage a hierarchy for the deliveries of a project in following manner: *Project_ID> DeliverySpecificationID> TeamID> teamDeliveryFile*
Interface/Exports	This component exposes an interface for saving, accessing and zipping team delivery files.
Methods	
Save file	
Name	save
Parameters	file, teamID, projectID, deliverSpecificationID
Return Value	True/false
Description	Saves the team delivery file in the file system
Data structure	File
Precondition	
Validity Checks	

Post conditions	The file is saved in the local file system of the server
Called by	Team delivery bean
Calls	File java class methods
Zip a set of team deliveries	
Name	toZip
Parameters	LinkedList<TeamDelivery>
Return Value	File
Description	Zip a set of team deliveries
Data structure	File
Precondition	The physical files are saved in the local file system, and the haven been moved to other path different from the register in the system database
Validity Checks	
Post conditions	The zip file is accessible to the user
Called by	Team delivery bean
Calls	

5.2.3 Persistence component



Component Design 13 Persistent component

Note: *The names of the entities can change according to my sql name restriction, p.e User can be a DBMS proper table, additionally the management of uppercase and lowercase of the DBMS.*

5.2.3.1 Project entity

Project Entity	
Classification	Entity
Definition	Is the entity that represents a project
Responsibilities	• Has all the project attributes specified in the database design
Constraints	
Composition	This entity has: • Delivery specifications • Teams • Types
Uses/ interactions	This interact with the following entities: • Delivery Specification • Team • Type
Resources	
Processing	Represents project data
Interface/Exports	
Methods	
Getters and setters	

5.2.3.2 Delivery specification entity

Delivery specification Entity	
Classification	Entity
Definition	Is the entity that represents a delivery specification
Responsibilities	• Has all the delivery specification attributes specified in the database

	design
Constraints	
Composition	This entity has: • Penalty • Team Deliveries • Type
Uses/ interactions	This interact with the following entities: • Team Delivery • Type
Resources	
Processing	Represents delivery specification data
Interface/Exports	
Methods	
Getters and setters	

5.2.3.3 Team delivery entity

Team Entity	
Classification	Entity
Definition	Is the entity that represents a team
Responsibilities	• Has all the team attributes specified in the database design
Constraints	
Composition	This entity has: • User • Team delivery
Uses/ interactions	This interact with the following entities: • User • Team Delivery

Resources	
Processing	Represents team delivery data
Interface/Exports	
Methods	
Getters and setters	

5.2.3.4 Team entity

Team Entity	
Classification	Entity
Definition	Is the entity that represents a team
Responsibilities	• Has all the team attributes specified in the database design
Constraints	
Composition	This entity has: • User • Team delivery
Uses/ interactions	This interact with the following entities: • User • Team delivery
Resources	
Processing	Represents team data
Interface/Exports	
Methods	
Getters and setters	

5.2.3.5 User Entity

User Entity

Classification	Entity
Definition	Is the entity that represents a user
Responsibilities	• Has all the user attributes specified in the database design
Constraints	
Composition	This entity has: • Profiles
Uses/ interactions	This interact with the following entities: • Profile • Team
Resources	
Processing	Represents user data
Interface/Exports	
Methods	
Getters and setters	

5.2.3.6 Profile entity

Profile Entity	
Classification	Entity
Definition	Is the entity that represents a profile
Responsibilities	• Has all the profile attributes specified in the database design
Constraints	
Composition	
Uses/ interactions	
Resources	

Processing	Represents profile data
Interface/Exports	
Methods	
Getters and setters	

5.2.3.7 Type entity

Type Entity	
Classification	Entity
Definition	Is the entity that represents a type
Responsibilities	• Has all the type attributes specified in the database design
Constraints	
Composition	
Uses/ interactions	
Resources	
Processing	Represents type data
Interface/Exports	
Methods	
Getters and setters	

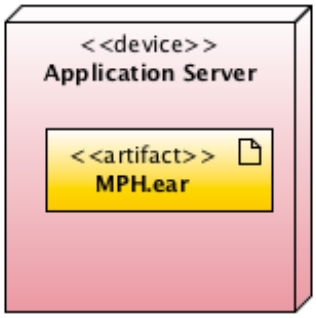
5.2.3.8 Penalty entity

Penalty Entity	
Classification	Entity
Definition	Is the entity that represents a penalty
Responsibilities	• Has all the penalty attributes specified in the database design
Constraints	

Composition	
Uses/ interactions	
Resources	
Processing	Represents penalty data
Interface/Exports	
Methods	
Getters and setters	

5.3 Runtime View

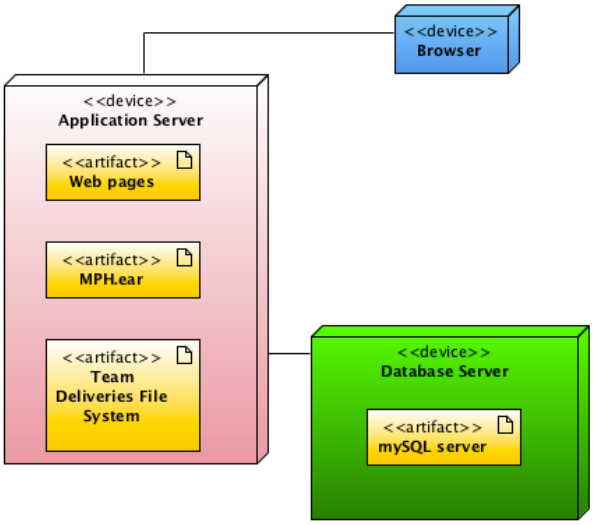
The software product will be release as MPH.ear, and can be deploy in a JEE application server.



Runtime View 1 Runtime View

5.4 Deploy View

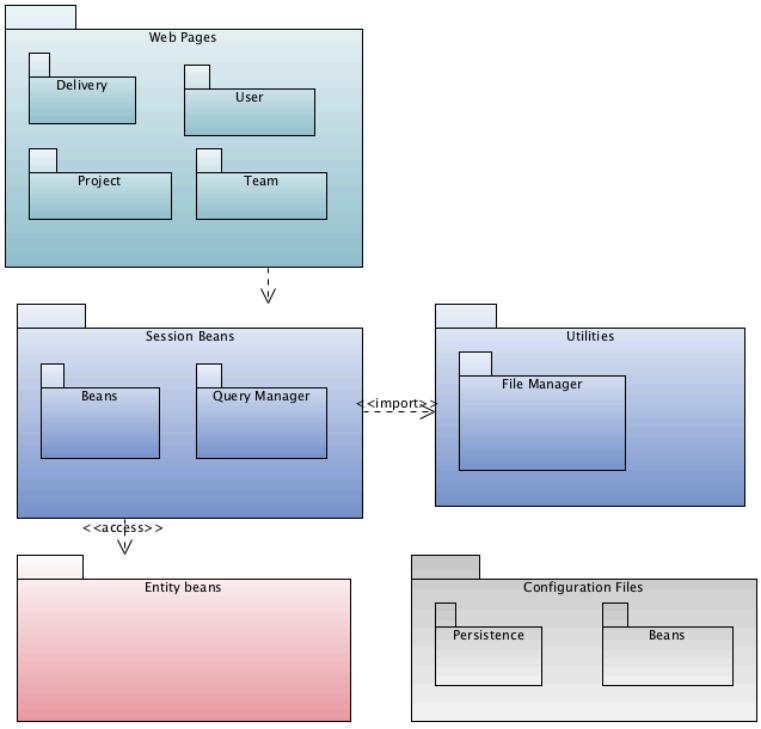
The following figure shows, the deployment view required by the software.



Deployment View 1 Deployment View

5.5 Module View

The following figure, shows an scheme for ordering the source code of the software.



Module View 1 Source Packages

6 Appendixes