REGIONE AUTONOMA DELLA SARDEGNA

SISaR

ENGINEERING

Servizio di Transizione

Titolo: XMPI/JBF/JAUF
Codice: AREAS_AP_01_JBF
Profilo: Analista Programmatore

www.eng.it

ALLEGATO 2: ESERCIZI

Main Bean - Exercise 1

We want to model a School Course, and we need to create the standard webpages to search, insert and edit a course.

Analysis

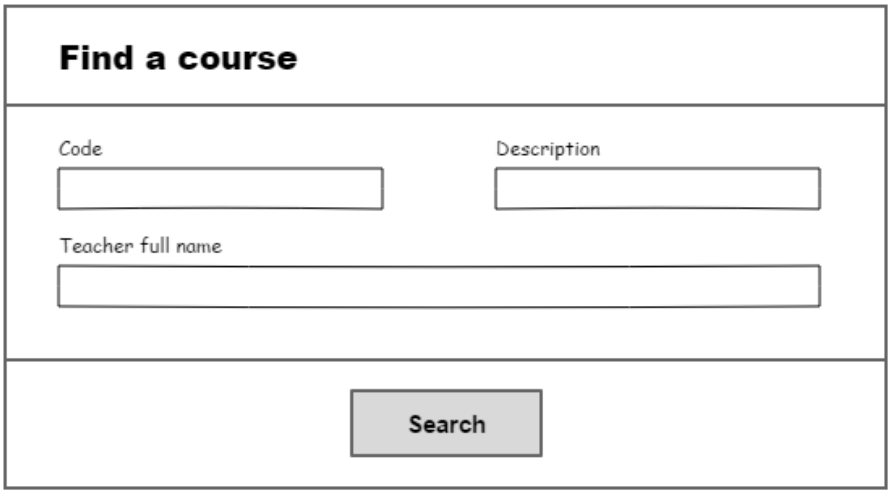
One single course needs to have these properties:

- A **code**, a unique identifier of the course (alphanumeric, max 11 chars)
- A **description**, a short description that says what the course is about (alphanumeric, max 256 chars)
- The **full name of the teacher**, one unique property that contains the name and surname of the teacher giving the course (alphanumeric, max 128 chars)
- A **start date**, when the course is supposed to start
- An **end date**, when the course is supposed to finish

Search page

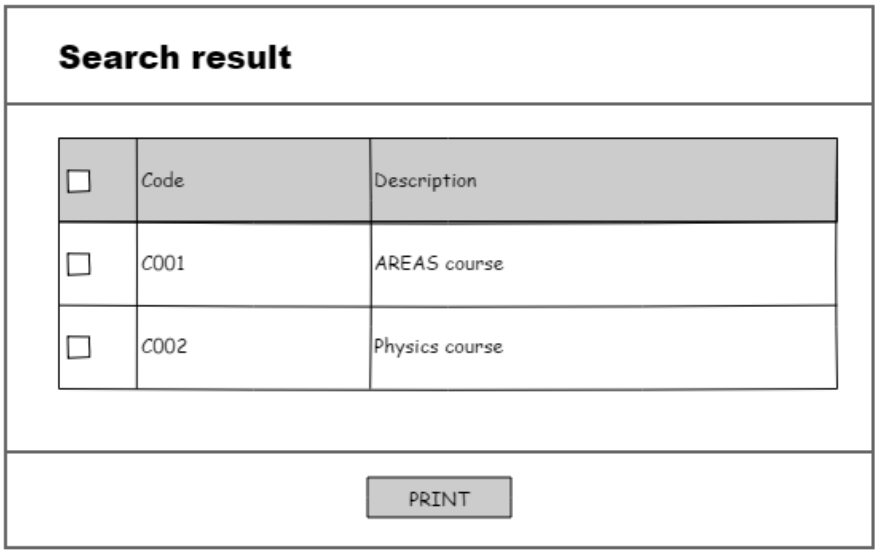
The user will be able to search a course using one or more of these filters:

- **Code**, the system will extract the only course who has exactly this code
- **Description**, the system will extract the course or the courses who have part of this filter in their description (like)
- **Teacher**, the system will extract the course or the courses who have part of this filter in their teacher name (like)



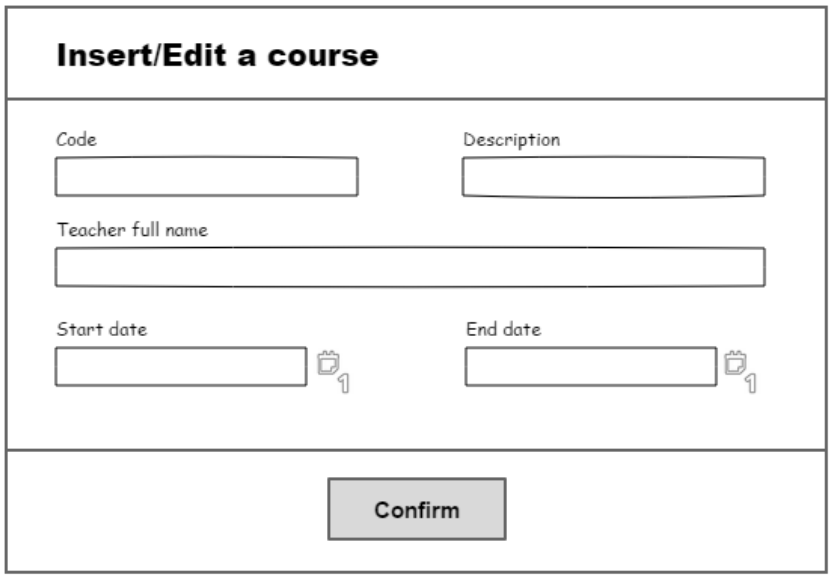
Result page

For each result, the system has to show **code** and **description** only, as shown in the picture below.



Insert/Edit page

The insert and edit page needs to allow the user to insert and modify all the properties of the course. No particular logics are required.

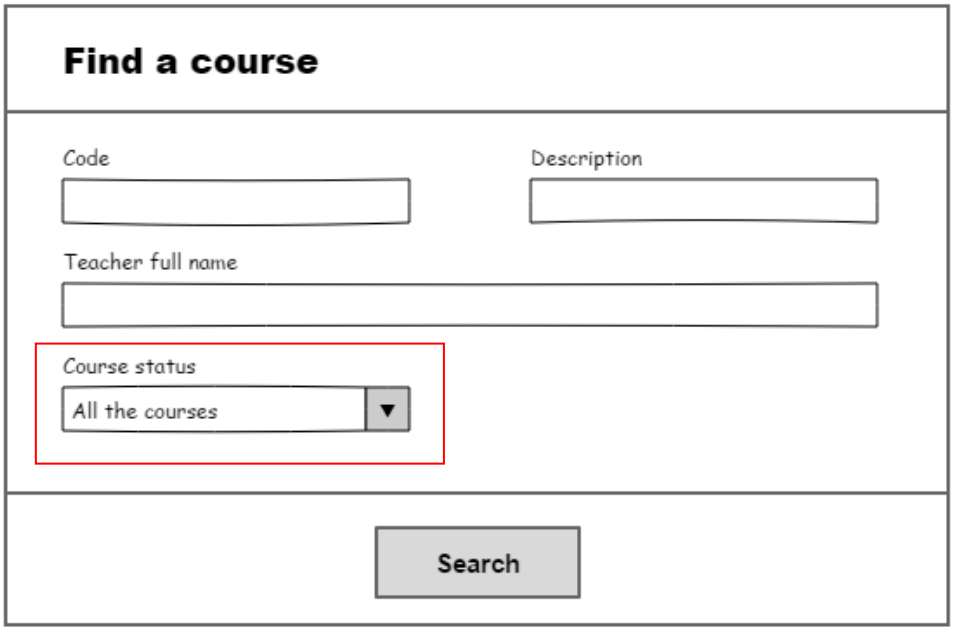


Exercise 1.1

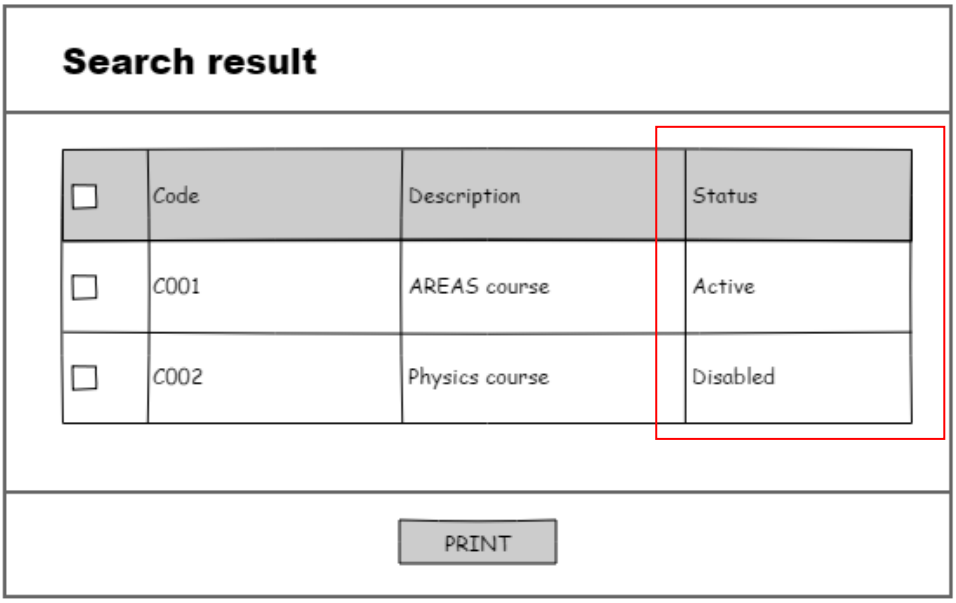
We add new functionality to the search page. A new filter “Course status” through with the user will be able to filter the courses based on their activation. In particular, the filter will be a combo with these options:

- **(T) All courses** (default option). Selecting this option, the system will not apply any additional filter
- **(A) Only active courses**. Selecting this option, the system will show only the courses that have no end date.

- **(D) Only disabled courses**. Selecting this option, the system will show only the course that have an end date

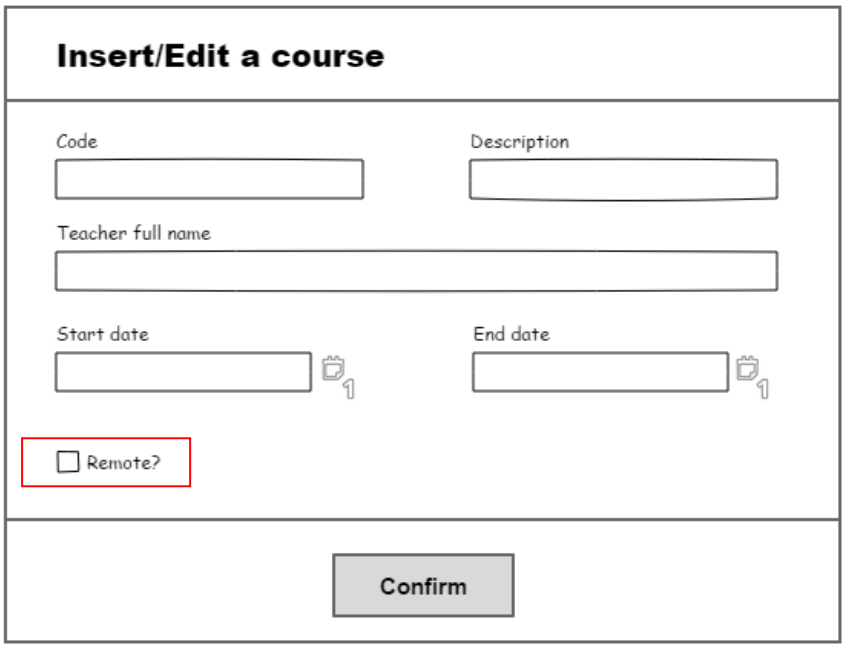


We need as well to modify the result page. In particular, we need to add a new column “Status” that will show the status of the course (Active or Disabled).



Exercise 1.2

We want to add a new editable property to a course. This property will be used to mark a course as a “Remote course”. For this reason, it will be a simple checkbox as shown in the picture below.



Visibility - Exercise 2

The goal of this exercise is to manage the visibility of the fields. In particular, we want to make the following properties mandatory:

- Code
- Description

Exercise 2.1

The goal of this exercise is to manage the visibility of the fields. In particular, if the user selects the option “Remote”, we need to make the following properties mandatory:

- Start date
- End date

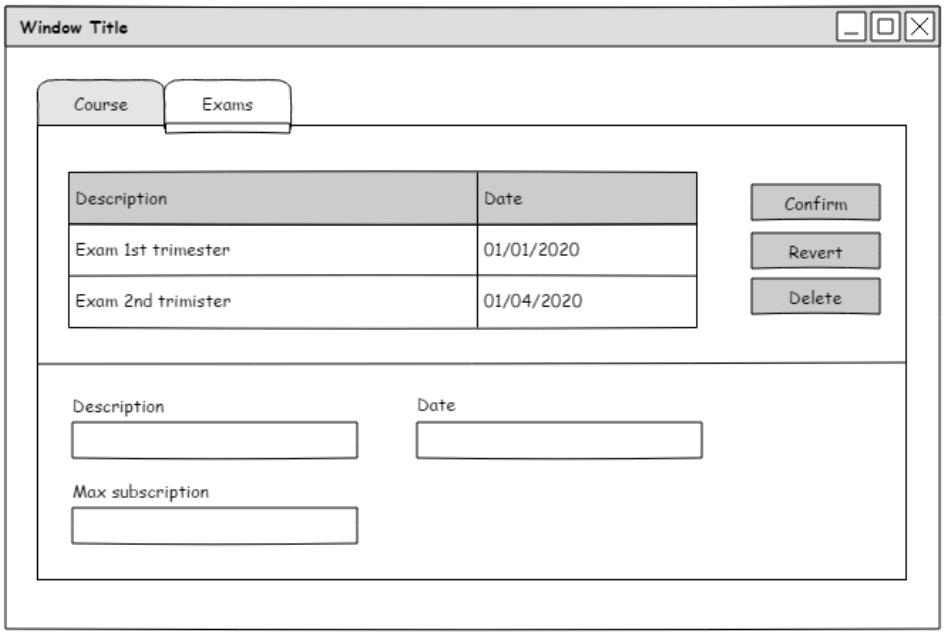
Detail Bean - Exercise 3

We want to add details to our course. This detail, in particular, will be the list of the exams scheduled during the year.

Analysis

Each exam must have the following properties:

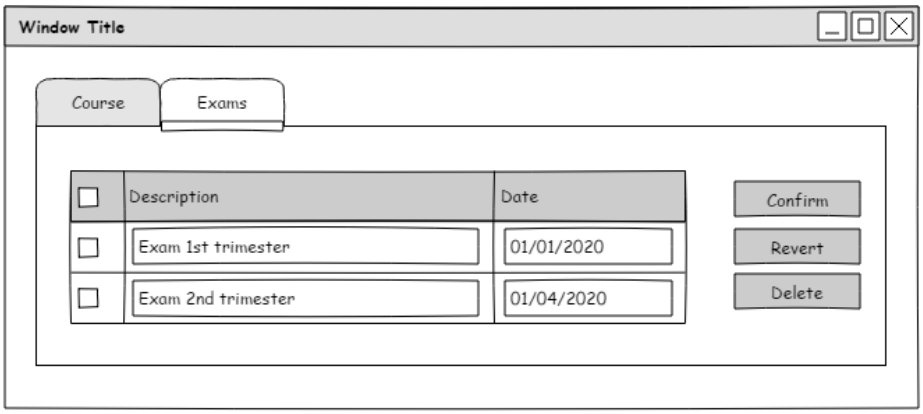
- Description of the exam
- Date of the exam
- Max number of participants allowed



To complete this exercise we also need to add an autogenerated id among the properties (SEQUENCE).

Exercise 3.1 (no JAUF)

The details above have really few properties. In this scenario, a view like the one of the previous exercise it's not user-friendly. The goal of this exercise is to transform this detail in an editable Grid.


Window Title
Course
Exams
Description
Date
Exam 1st trimester
01/01/2020
Exam 2nd trimester
01/04/2020
Confirm
Revert
Delete

Another Main Bean - Exercise 4

The goal of this exercise is to create another series of webpages from scratch. These webpages will allow the user to search, insert or edit a Student.

Analysis

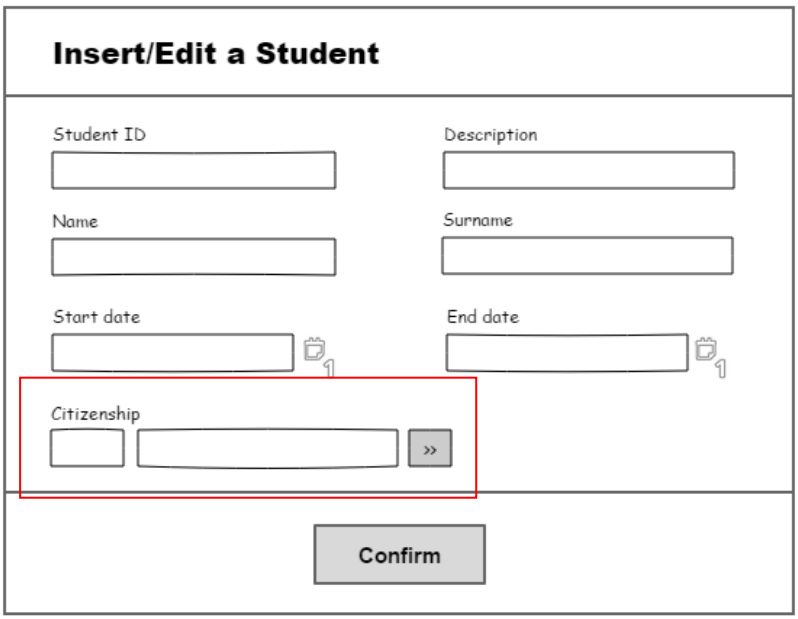
One Student needs to manage the following properties:

2. A unique ID auto-generated
3. The name of the student
4. The surname of student
5. A date representing the starting validity of its subscription
6. A date representing the end validity of its subscription

From the search page must also be possible to filter by ID or by part of the name or surname (like search). The result page needs to show only the ID and full name of each student.
We'll show all the fields in the Insert/Edit View and the ID field must be locked.

Lookup - Exercise 5

We want to improve the registry of Student (the main bean). Here we want to add the information about the citizenship of the student. We already have a registry of Citizenship in AREAS. Both lookup and table are called **SA_CITTADINANZA**.



Help:

1. **Add the field on your student table (FK to CT_ID of SA_CITTADINANZA)**
2. **Add the property in your Model (java, xml..eccc.)**
3. **Declare the lookup in the getObjectInfo**
4. **Add the widget in your View**

Exercise 5.1

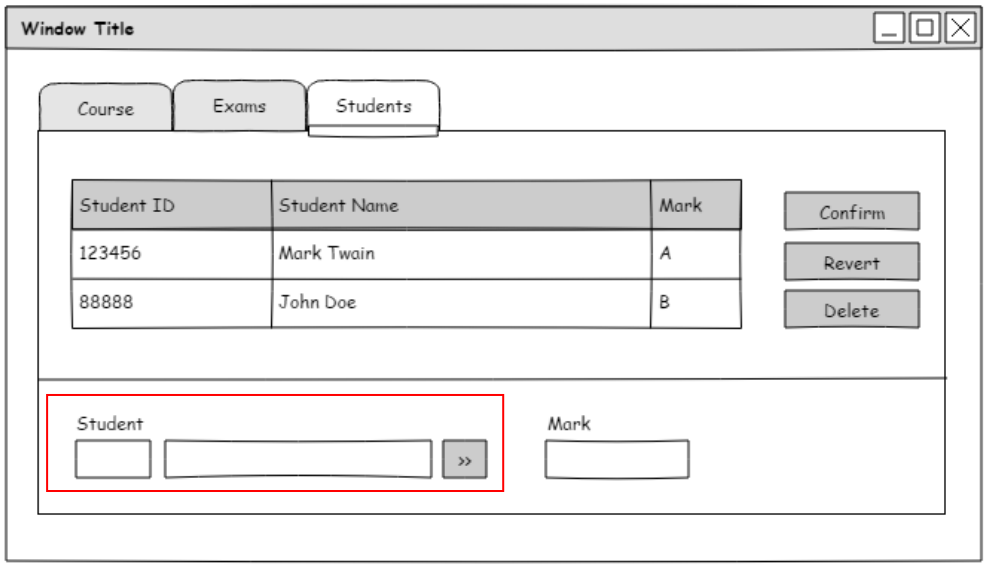
We want to add details to our course. This detail, in particular, will be the list of students, attending the course, and their marks. Since, in the previous exercise, we created the registry of Students, we want to simply refer a student insert using the registry.

Analysis

Each detail must have the following properties:

- A reference to the student
- The mark of the student

In order to have a reference to a student, we need to define a Lookup (we can call it STUDENTS).



SA_COURSE
- CO_CODE

SA_EXAMS
- EX_ID
- EX_CODE_COURSE

SA_STUDENT_COURSE
- SC_ID (pk sequnce)
- SC_CODE_COURSE (FK)
- SC_STUDENT_ID (FK)
- SC_MARK

SA_STUDENT
- ST_ID

Exercise 5.2

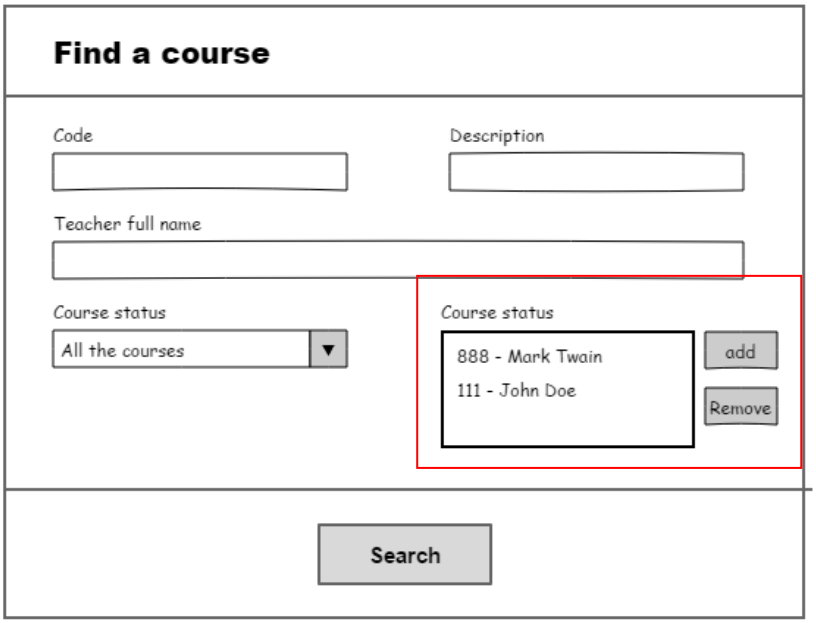
We want to show only the students currently “active”.

Exercise 5.3

Instead of using a simple “Value list”, we want to select our student using the “Search page” of the Student we created in the previous exercise (FIND2).

Exercise 5.4

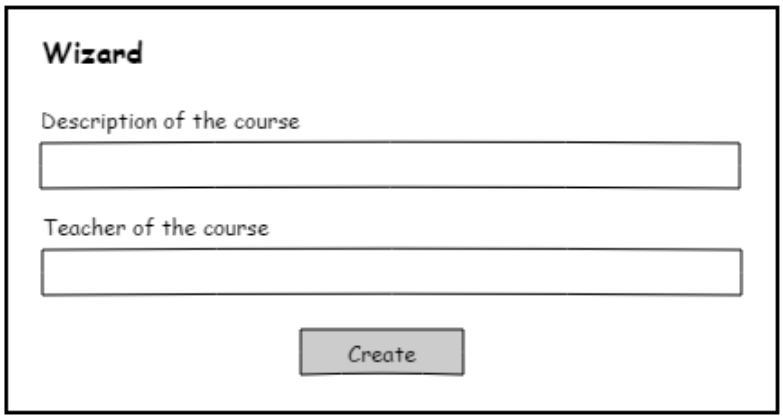
We want to add a new filter in the “Search page” of courses. This filter will allow the user to select one or more Students. The system will show all the courses that have all the students selected (FIND3).



Service Manager - Exercise 6

We want to create a new webpage, where the user can create a new course in a really fast way. In this wizard, the user will just insert the description of the course and the Teacher full name. The system will:

1. Automatically assign a code using this format “C_<incremental_number>” (SequenceManager.getNewValue(nomeSequence))
2. Set the start date on today

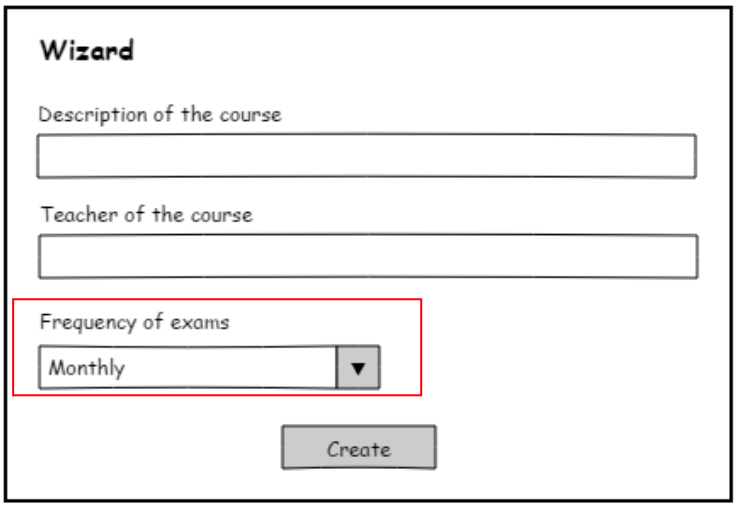


Exercise 6.1

We want to improve our wizard. Besides the fields above, we want to allow the user to automatically generate a calendar of exams on the course we’re creating. The user can select, trough a combo box, one of the following options:

- **Monthly**, the system will generate an exam starting the first day of every month
- **Trimester**, the system will generate an exam starting the first day of every 3 months
- **Semester**, the system will generate an exam starting the first day of every 6 months

The description of the exam will follow a format like “Exam <number of exams>”



Exercise 6.2

Starting from the previous exercise, we add the option “Annual” to the combo box. If this option is selected, we show up a new text field to indicate the date of the only exam we’ll generate.

Profiling – Exercise 7

We want to create a profile key. This key will establish if a user is allowed or not to use our wizard. The profile key needs to be an “Abilitazione”. If the user clicks the “Create” button and by profile key is not enabled to do it, an error message is shown.

Oracle name conventions

How to create a table

```
CREATE TABLE SA_STUDENT (
        ST_ID NUMBER(11),
);
```

How to specify a Primary Key

```
ALTER TABLE SA_STUDENT ADD CONSTRAINT PK_SA_STUDENT PRIMARY KEY(ST_ID);
```

How to specify a Foreign Key

```
ALTER TABLE SA_EXAS ADD CONSTRAINT SA_EXAMS_FK1 FOREIGN KEY (EX_CODE_COURSE)
REFERENCES SA_COURSE (CO_CODE)
```

How to create a sequence

```
CREATE SEQUENCE SEQ_SA_STUDENT
```