Test task definition

(BE/full-stack developer position)

Form-based data storing system

Objectives

The main goal of the test task is to find out if and how applicant can design and implement a simple working system according to given specification.

The system is a web-based application meant for performing common tasks on various not hard-coded data forms, such as adding, editing, removing a form and viewing a form data.

System definition

- Forms are described in editable templates that contain form name, types of fields with possible restrictions, and other necessary information.
- Each form template used in the system must be described in separate JSON file.
- System must allow user to add new form templates and delete form templates which are in use.
- Adding a form template can be performed by uploading a JSON file to the server (file upload using web-forms). Though it is not necessary to implement this, it's enough if system can just read JSON files from some folder.
- If JSON contains any errors and can't be parsed, it must be reported to user. Data entered to form must be stored in SQL database.

Form definition

Data form may contain any set of fields of the following types:

- Text (restrictions such as min and max length may be applied)
- Number (restrictions such as min and max value may be applied)
- Set of predefined values (one value may be chosen) e.g. "*Allow*", "*Deny*" or "*Mon*", "*Tue*, "*Wed*", ...
  In HTML form this type must be represented as a drop-down list.
- Boolean (check box)
- Date (Format: DD.MM.YYYY)

Each field in form must have a title.
If user enters data in wrong format, in must be reported to him.

System front page

System front page must contain the following controls:

- Drop-down list for selecting a form template to work with. On selecting some item from drop- down a table with entered data must be shown. This list must contain human readable form names, not just JSON file names.
- Table with entered data. Table columns must reflect form fields (with some exceptions, see below). There must be **Edit record** and **Delete record** button in each table row.
- **Add** button for adding a new record.

Examples:

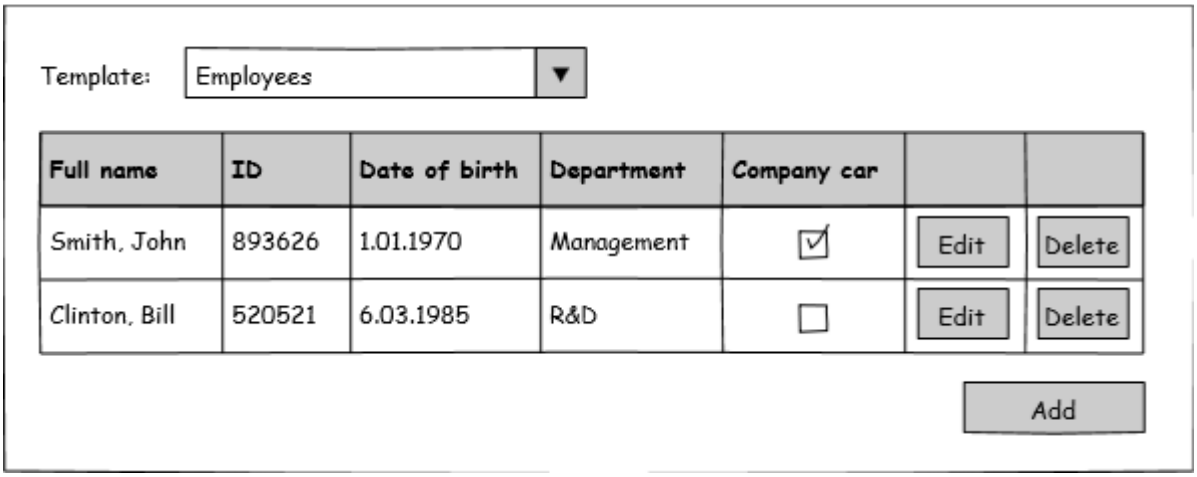

Figure 1: Table with data for selected template

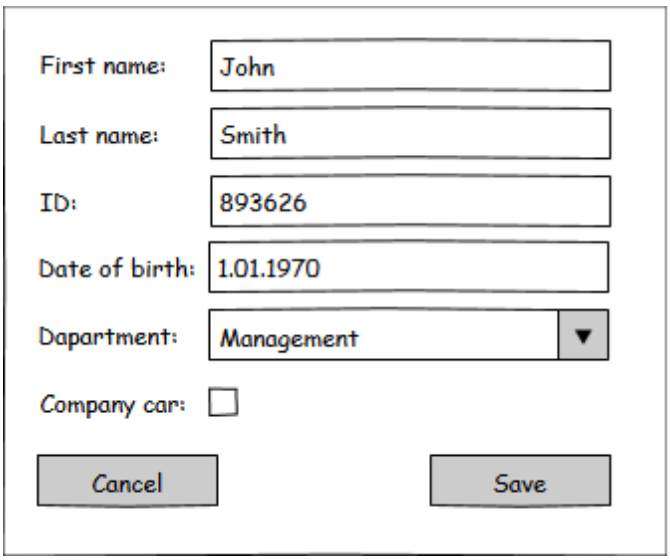

Figure 2: Form for entering data

Data in table may be a simple combination of entered fields. For example, if there are two fields in form

First name: John
Last name: Smith

But in table it may be presented as

Full name	or as	Full name	or as	Full name (ID)
Smith, John		John Smith		Smith, John (893626)

or in any other way.

I.e. the format of output data must not be hardcoded but must be configurable in form template. Even after the data is collected user must be able to change the format of columns of data table by making modification of a template.

User must have ability to sort a table by any of a column.

Test data for the task

There should be one JSON template created with the following fields:

- First name (text, min length 1 character)
- Last name (text, min length 1 character)
- Date of birth
- Internal company ID (10000-99999)
- Department (predefined list: “R&D”, “Sales”, “Marketing”, “Management”)
- Uses company car (Boolean)

All this fields must be shown in table in separate columns except for “First name” and “Last name” fields, which must be in one column “Full name” and have combined value “Last name, First name”.

NB! Again, this exception must not be hard-coded. JSON template must be editable even after some data is collected and must allow to add or to delete any kinds of fields’ combinations to the table. For example, admin may want to modify column “Full name” to have values like “First name, Last name”, or to have two separate columns, or to have column with combination of values like “First name, Last name (ID)”, or to have any other combinations with any other columns.

In other words – correlation between set of fields entered in form and columns / values displayed in table must not be hard-coded but must be somehow defined in template and must be alterable at any time.

Additional requirements

SQL-database structure must be able to handle all kinds of forms and must not depend on which forms are used. This means that in case if form was modified (for example, new field was added) database structure must remain intact.

The system must support any number of forms with any set of fields.

Task output

- All source files including 3rd party libraries if there are any in use. There must be clear distinction between all 3rd party libraries and applicant’s code.
- A script, command, or instructions for creating MySQL database tables.
- JSON file with form template as described.
- Some documentation (especially on JSON template format) will be a plus.

System must be easily deployed and tested. There will be a new JSON template created to test it with a system.

Some additional information

For making this system applicant may use any approach he or she wants, although applicant must show his or her skills in good software design, database design and coding.

In evaluation, the focus will be in how the non-hard-coded web form structure is implemented and how the data is stored to the DB. Hence the layout, error-checking, and validation of the data are not the most important part of this task.

System requirements

Frontend and backend may be implemented with any language/framework. The database should be SQL, preferably MySQL, MariaDB or PostgreSQL.