Design Document for ”CV Generator”

– SE 302 –

Principles of Software Engineering

submitted by

Berkin Öztürk
Ali Doğan Güllü
M. İrem Hatipoğlu
Ege Sevinç
Kemal Doğan
Yiğit İnan

December 8, 2022

Supervisor

İlker Korkmaz

Izmir University of Economics
Faculty of Engineering
Computer Engineering

Contents

List of Figures

1 Introduction

The CV Generator is a project that aims to store up to thousand CV files to be uploaded quickly to the system, to allow the person to input information about the person who has that CV, and to make this information easily accessible in the future by tagging. The application will be designed as a desktop application for Windows, and its language will be English. The Java programming language, SQL, and GUI will be used in the development of this application, so it will be very easy and fast for the user to use this application, which has a modern appearance, and to learn the operation after reading the user manual.

These satisfy the non-functional requirements listed below.

Non-functional Requirement 1: The application shall be run on Windows 10 Operating System devices and Intel processors.
Non-functional Requirement 2: The program shall be available in English.
Non-functional Requirement 3: The app shall be used easily after the user's manual reading.
Non-functional Requirement 4: The app shall be minimally designed with a modern look so the user will be familiar in less than a day.
Non-functional Requirement 5: The database should be able to support up to 1000 CVs.
Non-functional Requirement 6: The app shall be able to add a CV in less than 5 seconds.

2 Software Architecture

The CV Generator is a desktop application that will be run on Windows, and it does not need any other applications or services to work properly. This project is going to have a simple but useful structure. We will look at the operations that the CV Generator application can perform with various classes. In the end, a graphical user interface representation of the CV Generator application will be provided to visualize the project. To understand the structure of this application, it is best to start with the classes that comprise the CV Generator.

2.1 Structural Design

CV Generator relies on attribute tags that form its basis and specify properties. We use these tags as search tags so that we can easily find people of different abilities among different CVs. To handle these relationships, we created the CV, Tag, and CVManagementSystem classes, a total of 3 classes.

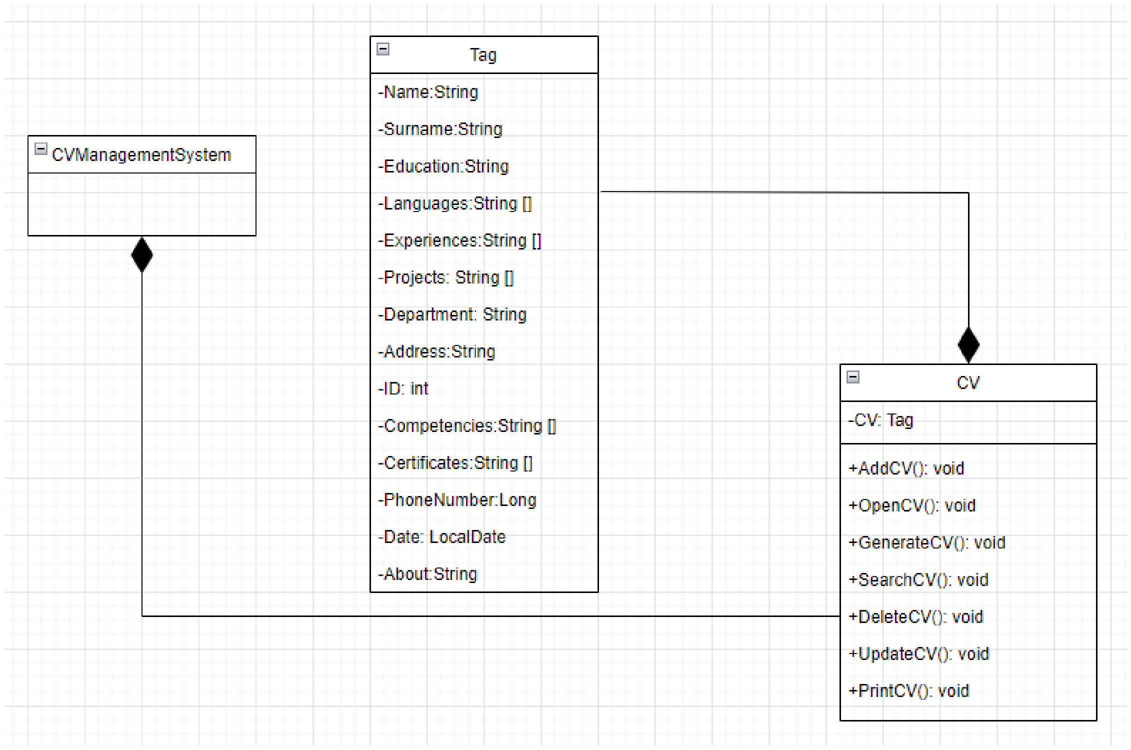

Figure 1: Class diagram for CV, Tag, and CVManagementSystem classes.

As shown in Figure 1, Tag class holds the name, surname, education, languages, experiences, projects, department, competencies, certificates, address, ID, phoneNumber, date, and about information about a CV.

CV class has one reference to the Tag class; the CV variable. With this relationship, we can keep Tag attributes together in the CV object.

A CV holder may have multiple qualifications or competencies in some areas. That's why languages, experiences, projects, competencies, and certificates are kept in an array.

For example, let's assume that the user creates a new CV object.

```
berkin = CV('Berkin, 'Öztürk, 'Bachelor', languages = empty, experiences = empty,
projects = empty, 'Computer Engineering', competencies=empty, certificates =
empty, 'Alsancak sk. no:29', 1, 5301781925, 12/5/2022, 'I am an engineer');
```

This object can be saved to the device and database as a stand-alone CV.

Using this approach, CVs can be added to examine the other person's CVs.

When we look at our CV class, we see that it contains AddCV, OpenCV, GenerateCV, SearchCV, DeleteCV, UpdateCV, and PrintCV functions.

These functions, which are kept in the CV class, are used to add a CV file, open a CV file, generate a CV, search for a specific CV, delete the CV, update the CV and print the CV respectively.

CVManagementSystem is the main class. We design our main menu and call our functions here. Apart from that, the class does not have any object or function.

These classes and relationships between these classes meet Functional Requirement 1, Functional Requirement 2, Functional Requirement 3, Functional Requirement 4, Functional Requirement 5, Functional Requirement 6, Functional Requirement 7, Functional Requirement 8, and Functional Requirement 9 which have been defined in the requirements document.

Functional Requirement 1: The user shall be able to add a CV to the System.
Functional Requirement 2: The program shall be able to show, add, delete and search functionality with buttons.
Functional Requirement 3: Name, personal information, skills, education, and projects, must be completed by the user.
Functional Requirement 4: The user shall be able to search for a specific CV.
Functional Requirement 5: The user shall be able to edit a specific CV.
Functional Requirement 6: The user shall be able to delete a specific CV.
Functional Requirement 7: The user shall be able to generate a CV template.
Functional Requirement 8: The user shall be able to tag CVs for easier navigation.
Functional Requirement 9: The user shall be able to print a specific CV.

2.2 Behavioural Design

While the structural design of the classes can store the CV in the database, the following methods are required to add functionality to the software.

Tagging Cvs and Searching
Functional requirement 2 states that users shall be able to search CVs with given tags with the help of a button. In our program CV class has-a tag and the tag class has attributes such as name, surname, language, etc. These tags can be searched via the textbox in our application and the click of the ”search” button. The user can search for any of these string tags, not just the name. This button and the textbox are created in our Java Swing GUI. With their respective classes and objects to provide the proper functionality that we desire.
This functionality will meet Functional Requirement 2.

Functional Requirement 2: The program shall be able to show, add, delete and search functionality with buttons.

As functional requirement 4 states the user shall be able to search for a specific CV. We will accomplish this by the use of our ”tags” so that our program does not have to go through the entire CV file. After the creation of tags for a CV, our ”Search” method will go through the attributes of our tag class and will match these string attributes with the ”Contains()” method so that we don’t have to type the exact tag such as ”Software Engineer” only ”Software” will be enough for us to find it.
This functionality will meet Functional Requirement 4.

Functional Requirement 4: The user shall be able to search for a specific CV.

And finally, our functional requirement 8 states that the user shall be able to tag CVs for easier navigation. We will achieve this with the Has-a relationship between CV and Tag. So the user will create tags with constructors for a CV and that CV may have multiple tags. Such as ”Software Engineering” and ”PHP”. These correspond to different attributes of tag class so we can easily say that all these fields are not required. There could be only a name or only Department or there could be no tag at all for a CV. We want to make our CVs highly customizable with these tags

and with this, the CV search will be easily done through our app. For example, if the user wants to see all computer science students in the system; he/she can easily achieve this with search functionality that goes through the attributes of CVs.
This functionality will meet Functional Requirement 8.

Functional Requirement 8: The user shall be able to tag CVs for easier navigation.

Adding CVs to Database
Functional Requirement 1 states that the user shall be able to add a CV to the System. The CV Management System database stores all CVs. However, this requires a database system connection. Using the Text Fields with Swing UI, the user shall enter the information specified by the tags of the person who sent the CV. The information entered in these Text Fields shall be saved to Database. Also, to see the other information specified in the CV; the CV shall be added as a PDF to the system then the database shall store it as a PDF.
This functionality will meet Functional Requirement 1.

Functional Requirement 1: The user shall be able to add a CV to the System.

Deleting CVs from Database
Functional Requirement 6 states that the user shall be able to delete a specific CV from the database. The user shall be able to delete a registered CV from the database by using the list that appears after searching for a CV. In addition, the user shall have the option to delete a CV selected from the list while observing it. Delete buttons shall be added for handling this function. The selected CV shall be deleted from the database when this button is pressed.
This functionality will meet Functional Requirement 6.

Functional Requirement 6: The user shall be able to delete a specific CV.

Generating a CV
Functional Requirement 9 states that the user shall be able to generate a CV template. While uploading a CV to the system on the CV adding page, the user shall fill in the TextFields and set tags, and the user shall also import the CV document to the database. If the CV document is not mentioned in this part, it is left blank,

the system shall automatically generate a new CV from a template with the HTML extension. Then this HTML file shall be imported into the database. The layout of this template will be determined by the system and shall be fixed. The HTML template shall be personalized by the system according to the information entered in the TextFields.
This functionality will meet Functional Requirement 7.

Functional Requirement 7: The user shall be able to generate a CV template.

Printing CVs from Database
Functional Requirement 9 states that the user shall be able to print a specific CV. User shall be able to print the CV in PDF format with a "Print" button added using Swing UI when examining a CV. When the "Print" button is pressed, it shall export the stored PDF document from the database and direct it to the print process for that document.
This functionality will meet Functional Requirement 9.

Functional Requirement 9: The user shall be able to print a specific CV.

Updating a CV
Functional Requirement 5 states that the user shall be able to edit a specific CV. The user shall be able to edit Tags that are exported from the database to Text Fields. Then with the "Update" button that was added using the Swing UI, these edited Tags shall be saved to the database again as they have been edited.
This functionality will meet Functional Requirement 5.

Functional Requirement 5: The user shall be able to edit a specific CV.

3 Graphical User Interface

Underneath, there are a few sketches to better understand how to design the GUI. GUI design is simple to understand and straightforward. In the implementation, there might be changes or fixes in the application, and the GUI shall be changed accordingly.

Non-Functional Requirement 4: The app shall be minimally designed with a modern look so the user will be familiar in less than a day.

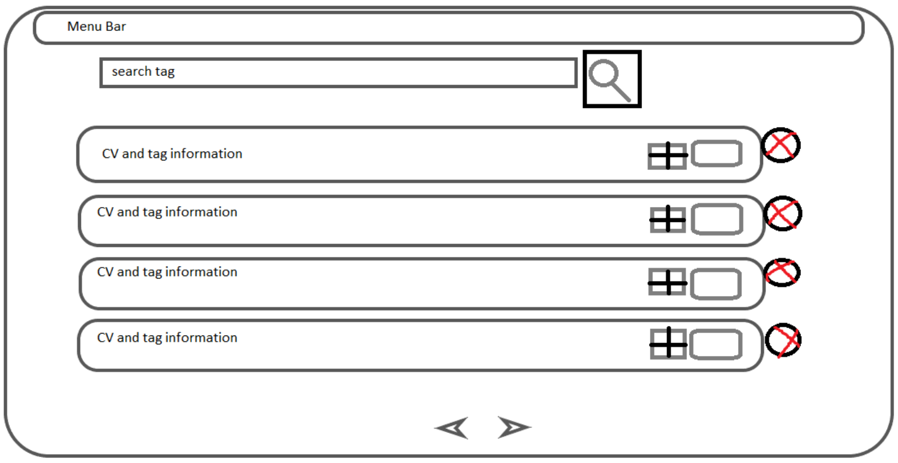

Figure 2: First window

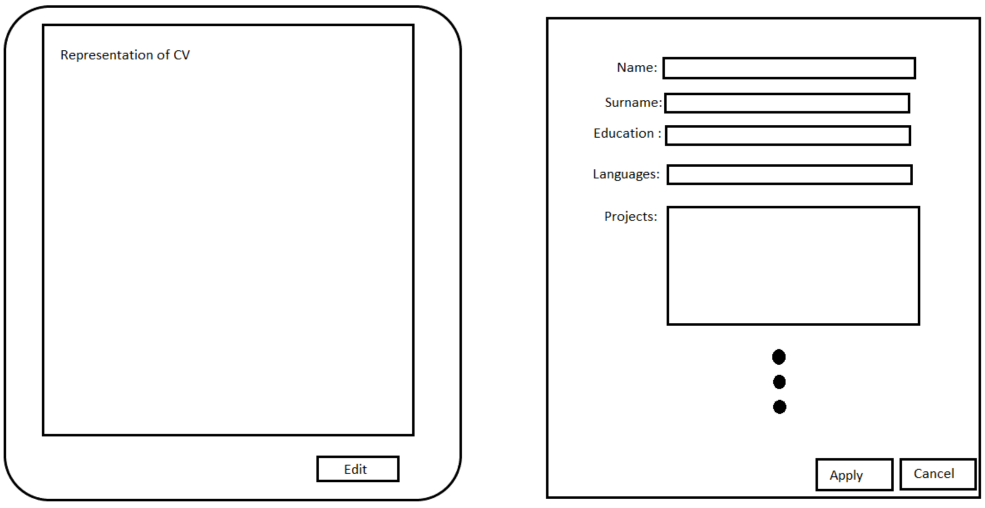

Figure 3: Visualizing and editing inside HTML content

When the user opens the program, the user shall visualize CV's name up to 10 and their related tag information in ascending order of last edited time. If the user wants to see more CVs, they will press the right button to go further. By clicking the right button on the search list user shall be able to visualize CVs. If the user generates a CV within the software, an HTML representation window will show up. Generation of CV shall be pressing the menu bar at the top. It will open another window and type out their categories and related information. The Import CV button will open the directories window. When the user shall save his/her file document then the user will add tags in the second window. For convenient usage for appending tags, the user can directly press down the right plus button.

Navigation will be done when the user types the related tags and clicks the button. The non-matched situations will inform the user as a pop-up warning. The UI will have a Menu bar with the following actions:

Tools

- Import CV
- Print CV
- Generate CV

Help

- Help

In the help menu, there shall be a user manual. It will not be online therefore user reach within the software.

Non-Functional Requirement 3: The app shall be used easily after the user's manual reading.