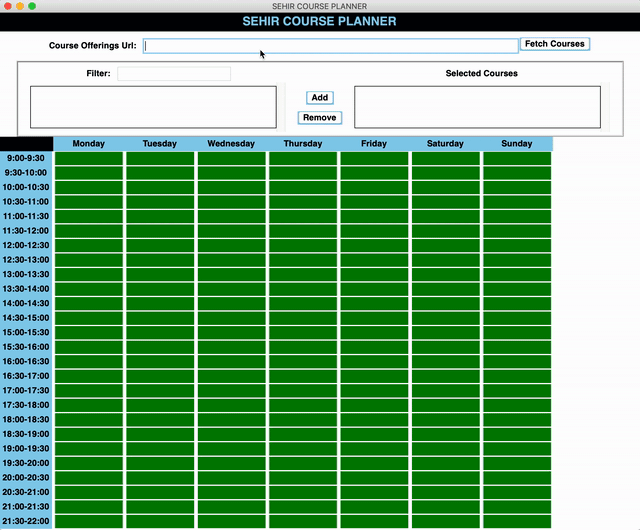
**ENGR 102 – Programming Practice**

**Mini Project 5**

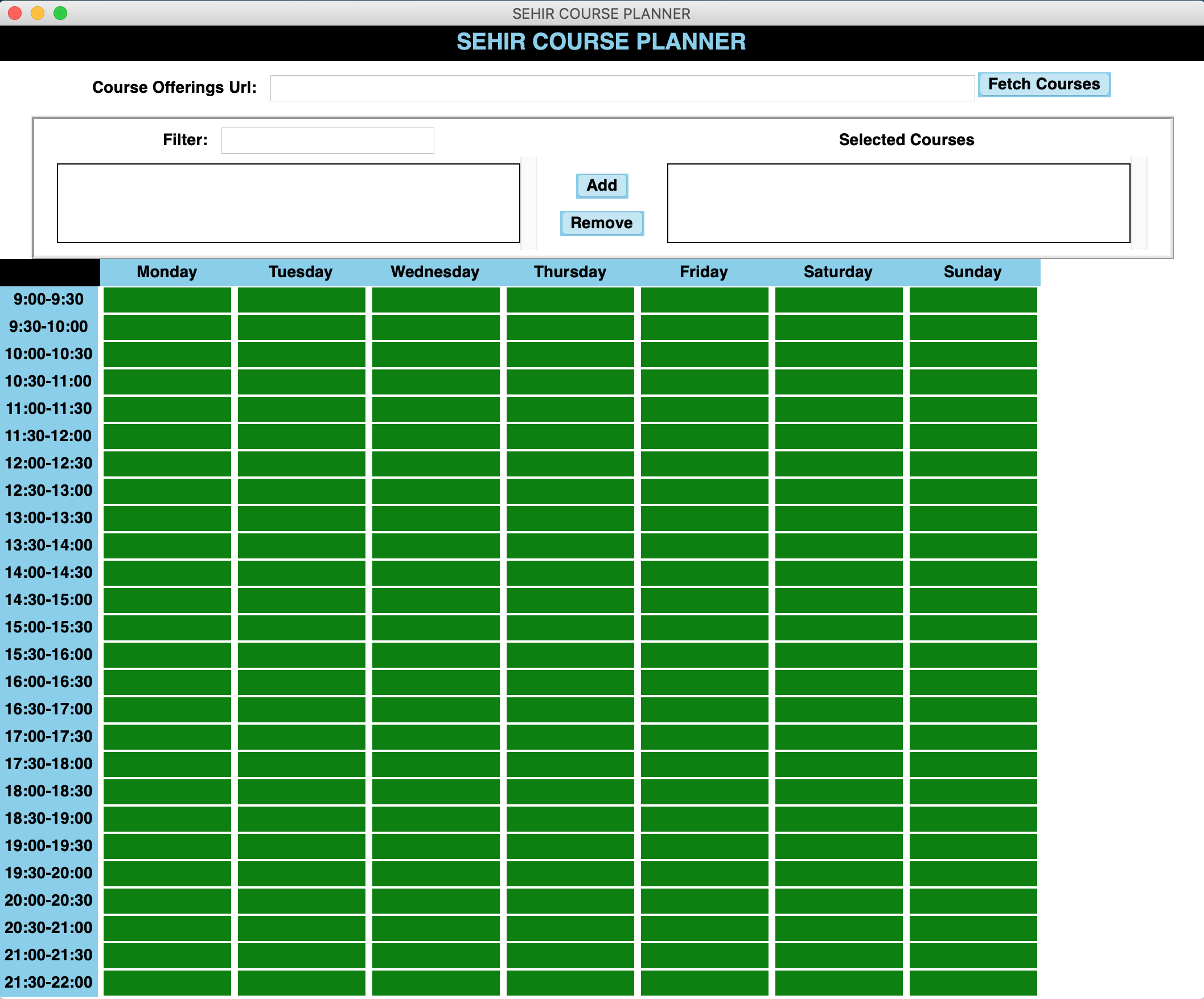
**Spring 2020**

***May 28, 2020 by 5 pm***

For this project, you will build a schedule maker for Sehir’s courses. Given a URL of the offered courses for a semester, your program will fetch the offered courses data and allow the users to make a schedule. The following is a **GIF** that shows how the program should function. For more details keep reading.



**How it should look like:** Your graphical user interface should look like the following one.

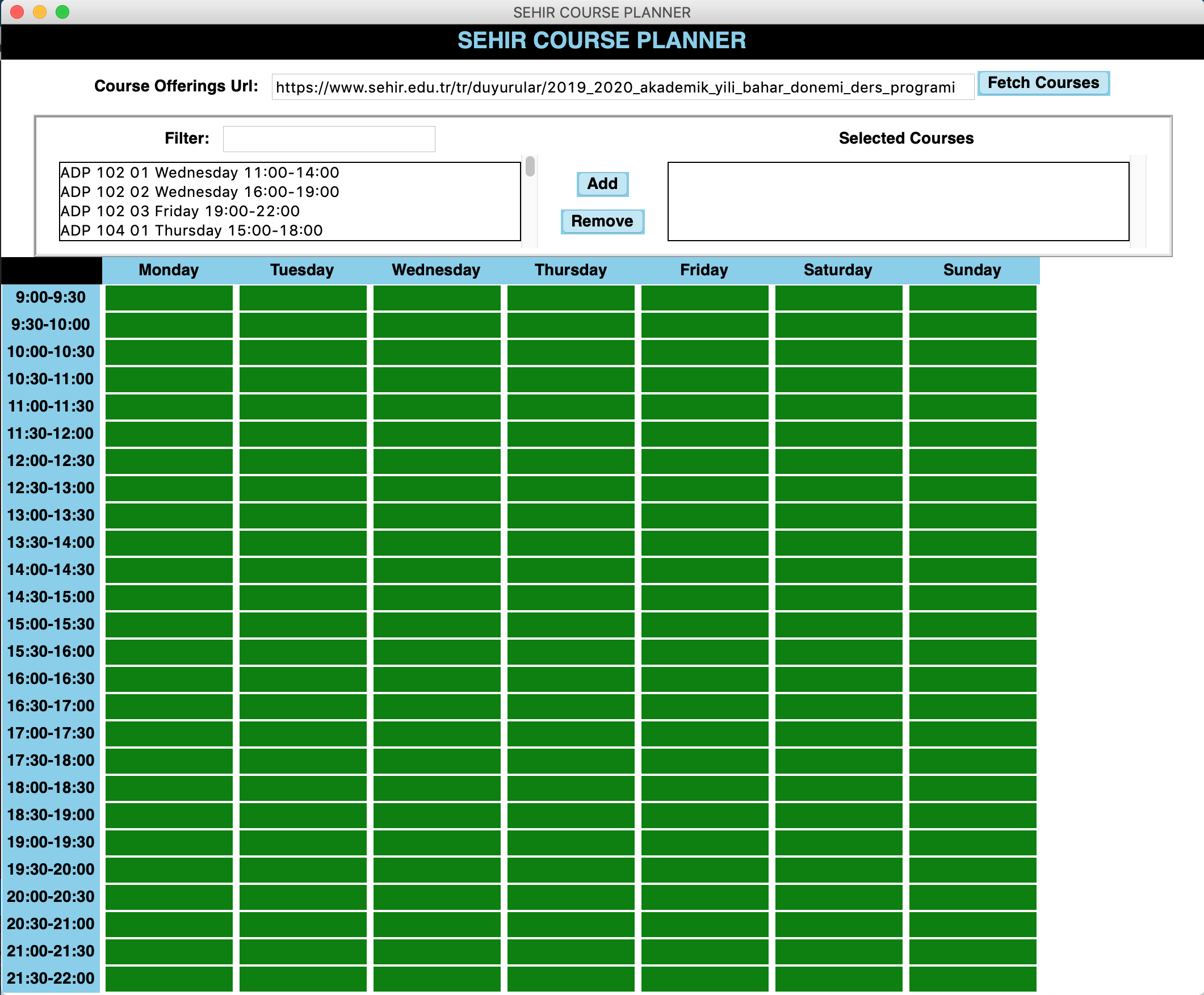
****

**How it should work:**

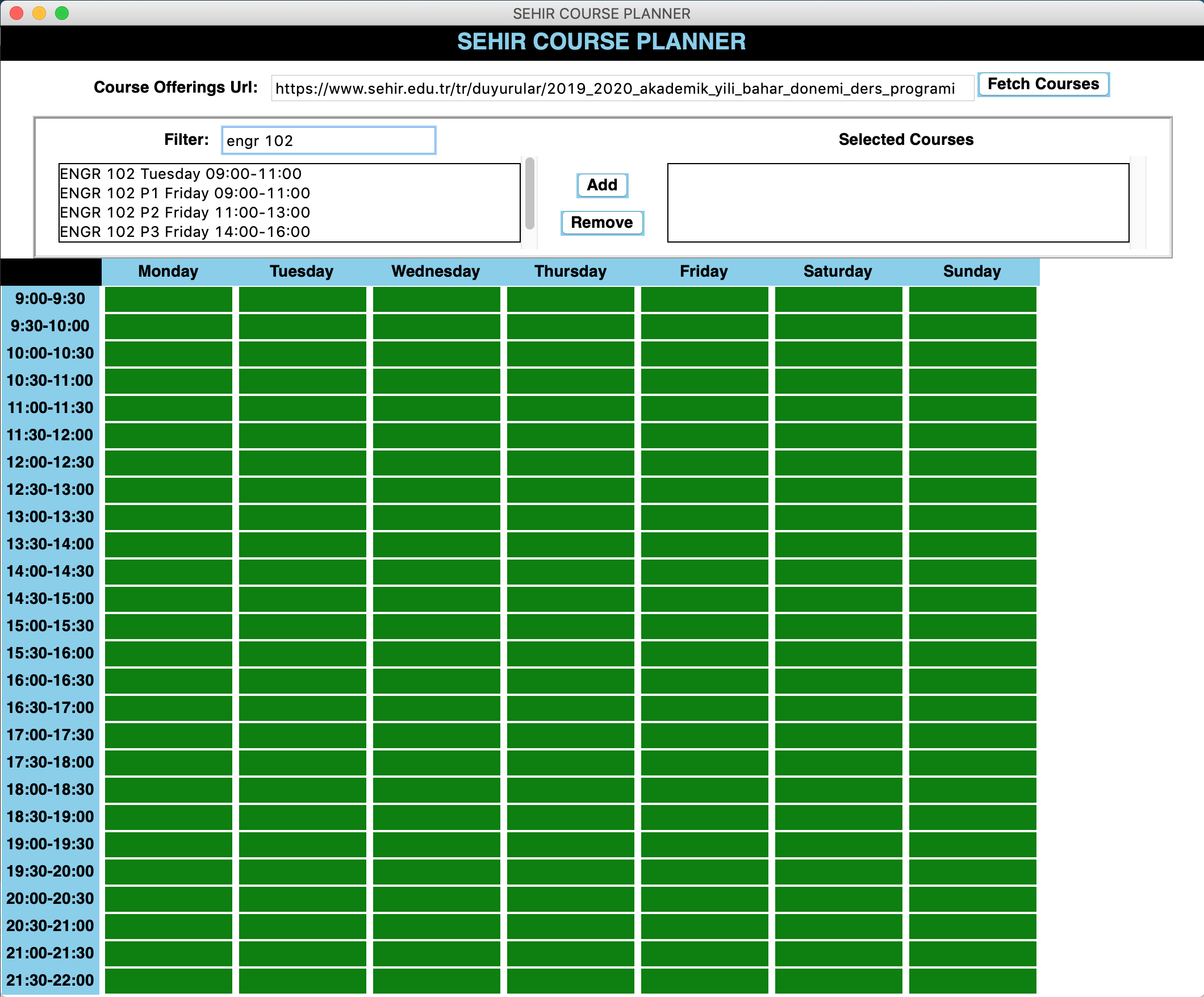
1. When the app is opened, the user will then provide a URL of the official schedule, then press **Fetch Courses.** When the button is clicked, the **listbox** on the left will be populated with the courses found on the schedule found at the given URL. Please make sure that the URL entry is populated with the following URL by default at the start of the program:

[**https://www.sehir.edu.tr/tr/duyurular/2019\_2020\_akademik\_yili\_bahar\_donemi\_ders\_programi**](https://www.sehir.edu.tr/tr/duyurular/2019_2020_akademik_yili_bahar_donemi_ders_programi)

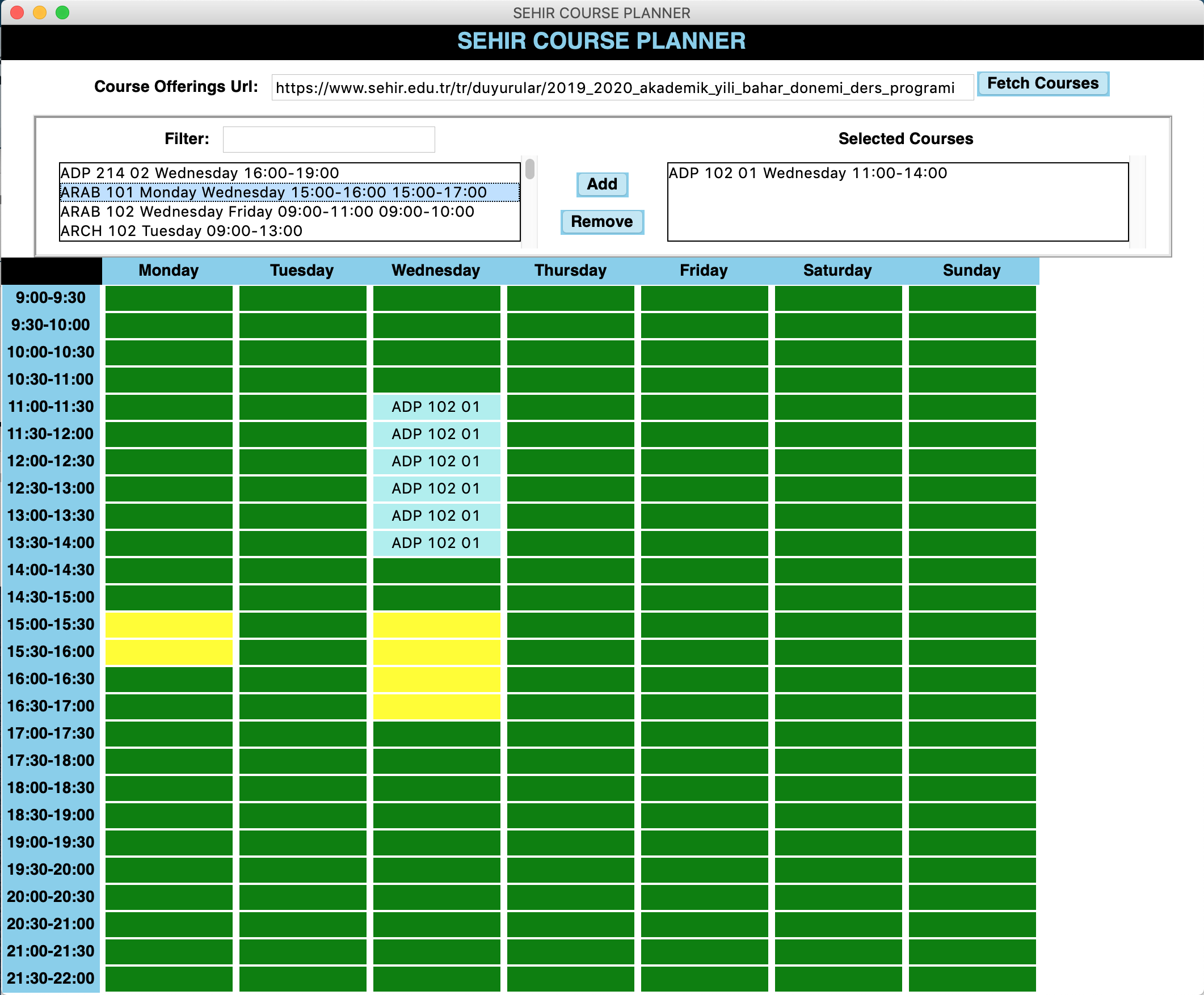
* 1. Make sure the list box contains **ONLY** the courses that have information about both **days** and **times** because some don’t. Also, make sure that courses are sorted by their course codes.



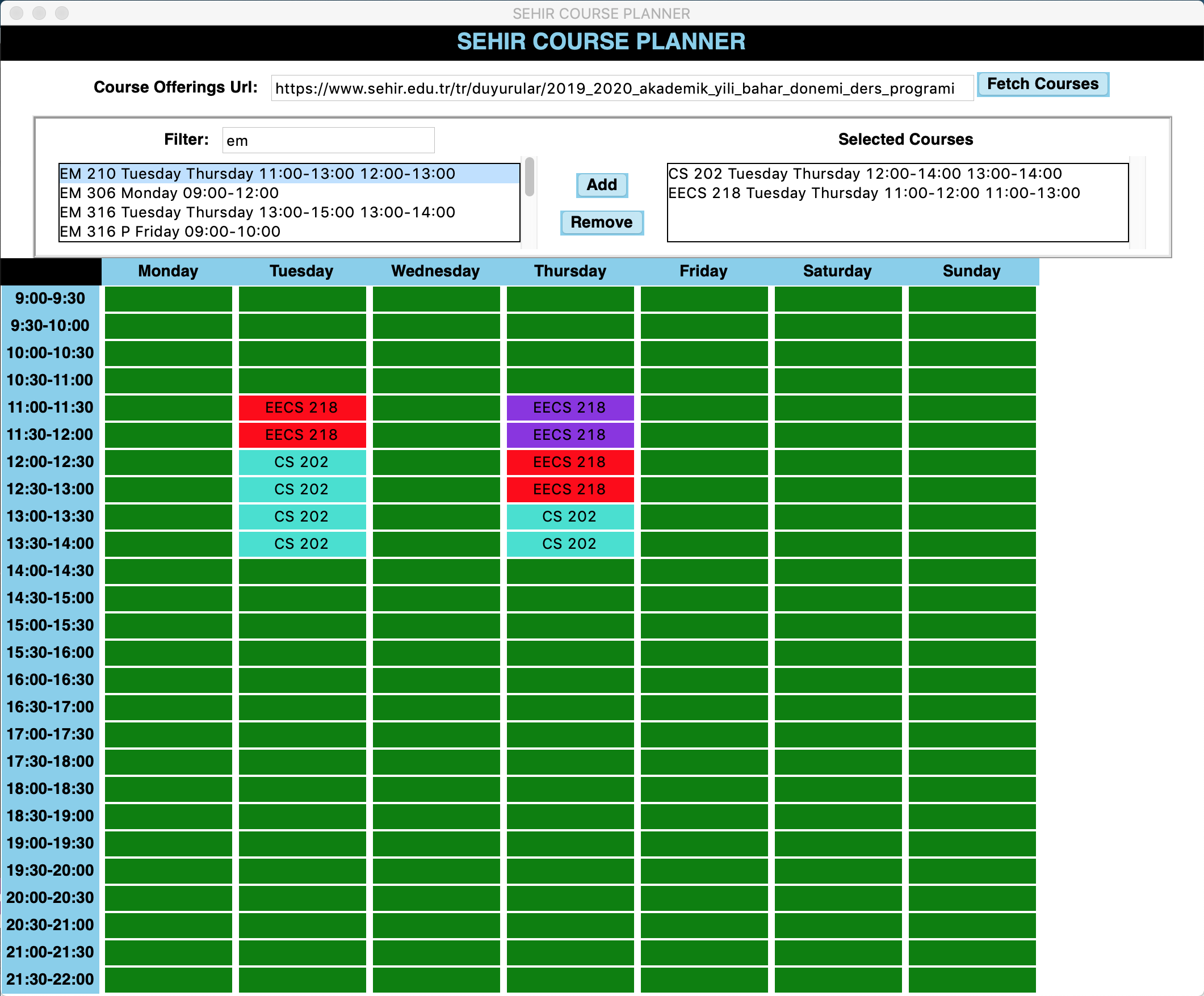
1. When the user writes something in the **filters** **Entry**, you should **ONLY** show the courses that **contain** the entered **text** or query. Check the figure below.

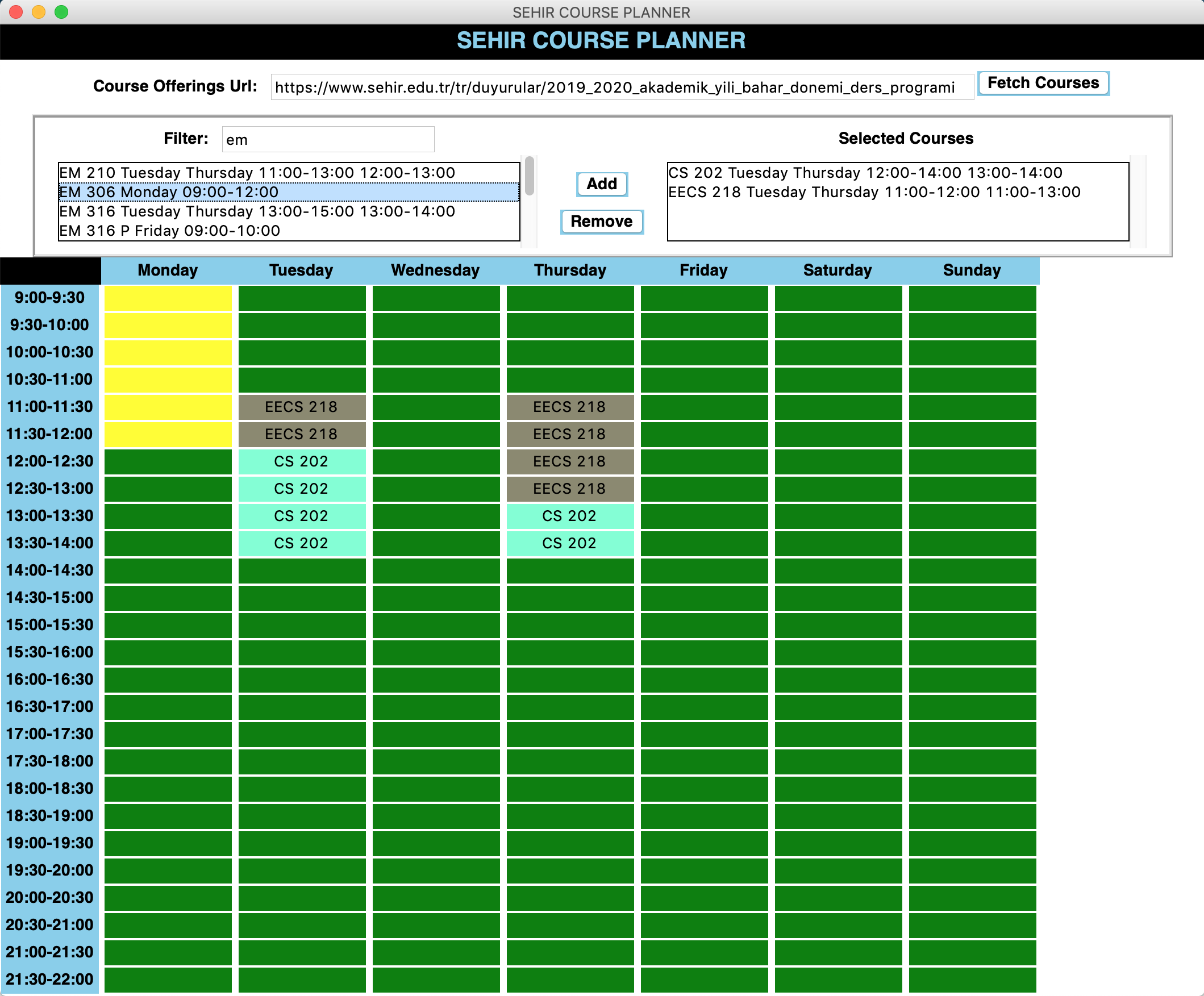


1. A user should be allowed to select **one course at a time** from the listbox on the left. When a course is selected:
   1. The corresponding cells on the timetable will be marked as **yellow** **if the selected course does not clash with any of the added courses** (that are placed in the right listbox).



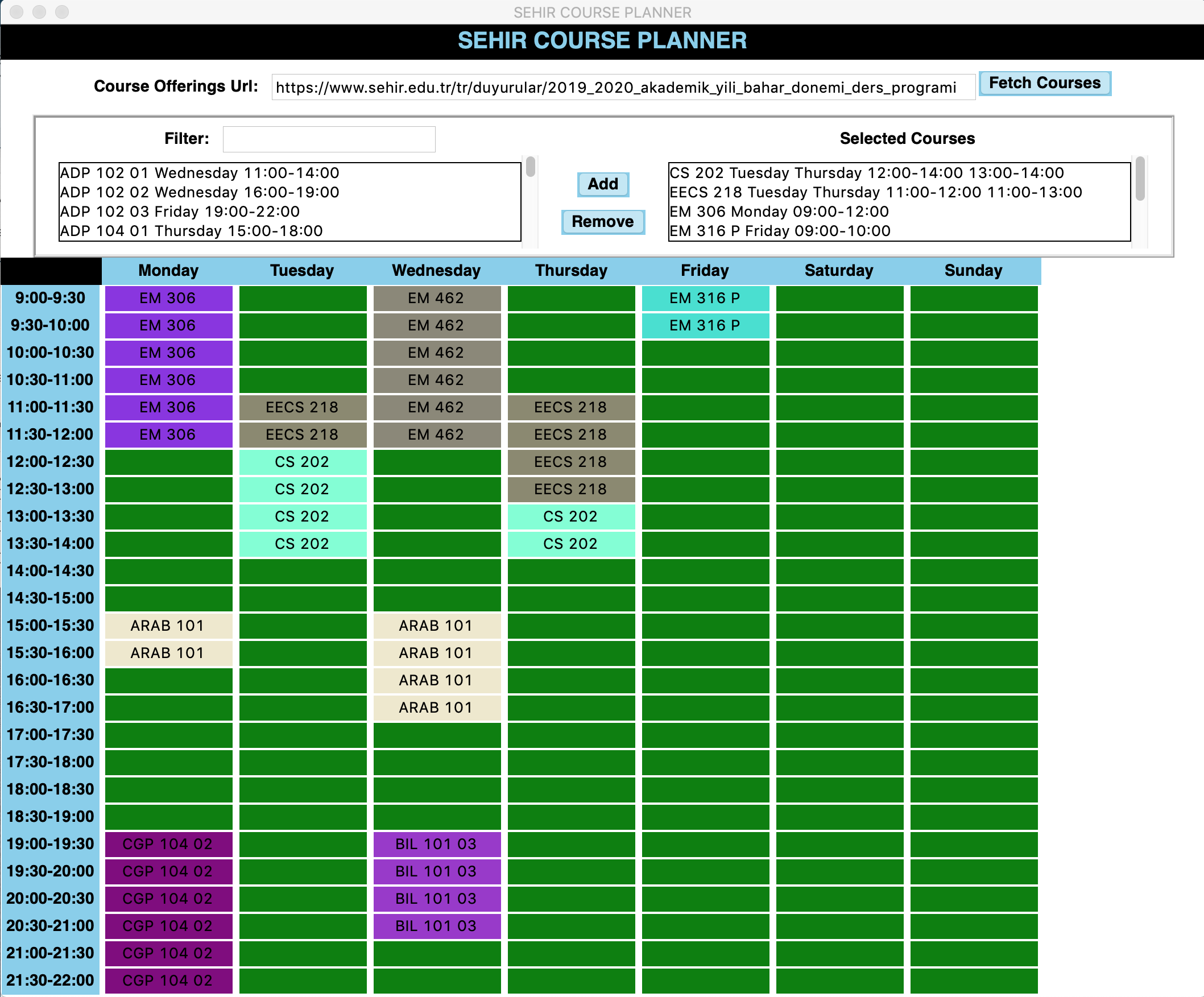
* 1. If the selected course clashes with any of the added courses, **ONLY** the cells where the clash happens should be red. Also, when you select any other course, **the red cells should be resetted to its previous state**. Check the following two figures.



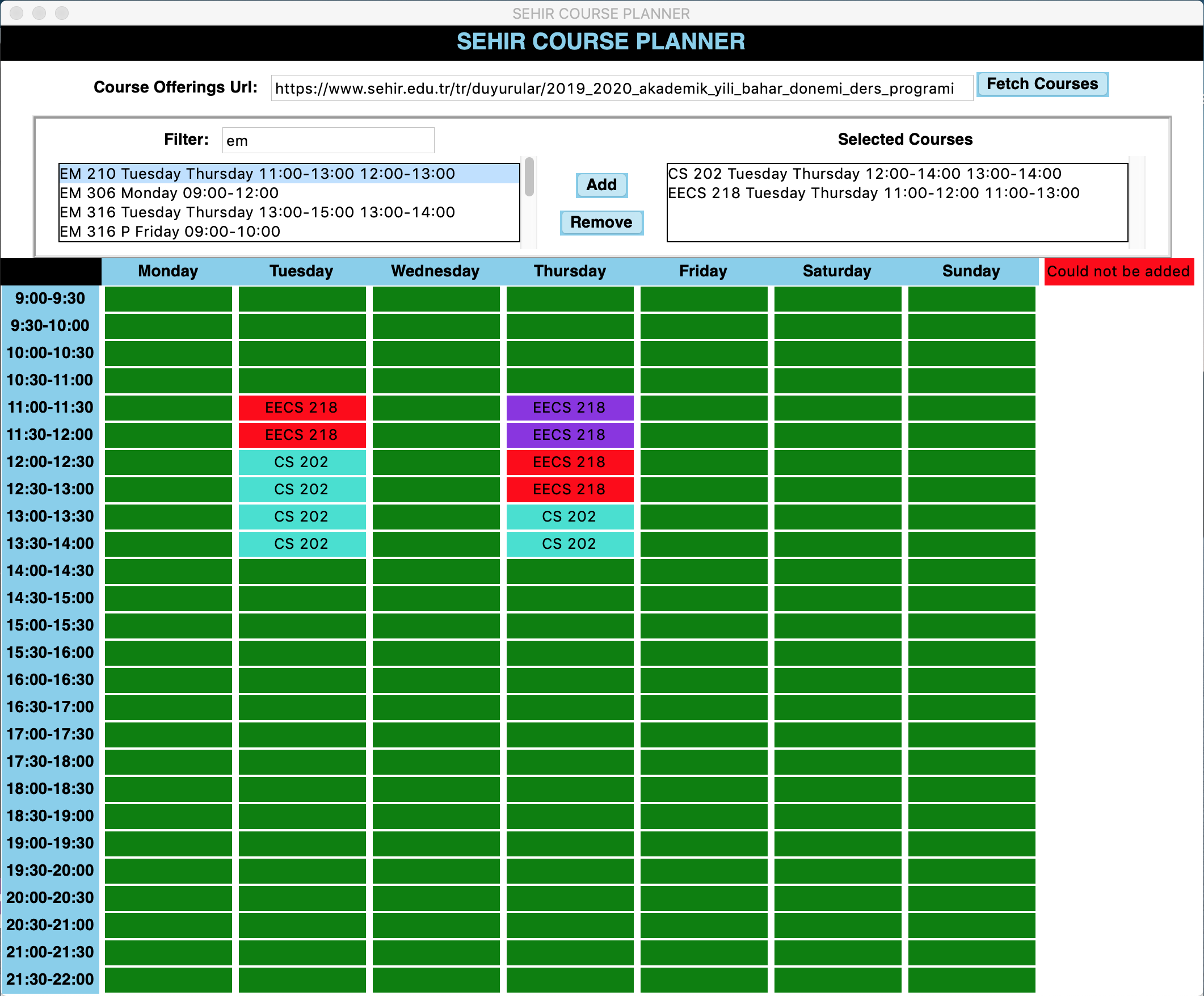


1. When a user clicks on the **Add** Button after selecting a course:
   1. If the selected course **does not clash**, the course will be **added to the listbox on the right.** And the corresponding cells of a course will be colored with a random color, and the **code** of the course should be written in every cell that the course schedule spans.
      1. **NOTE:** Each added course should be uniquely colored. You may choose from the following colors. You may assume that the number of added courses will not exceed the number of colors here.

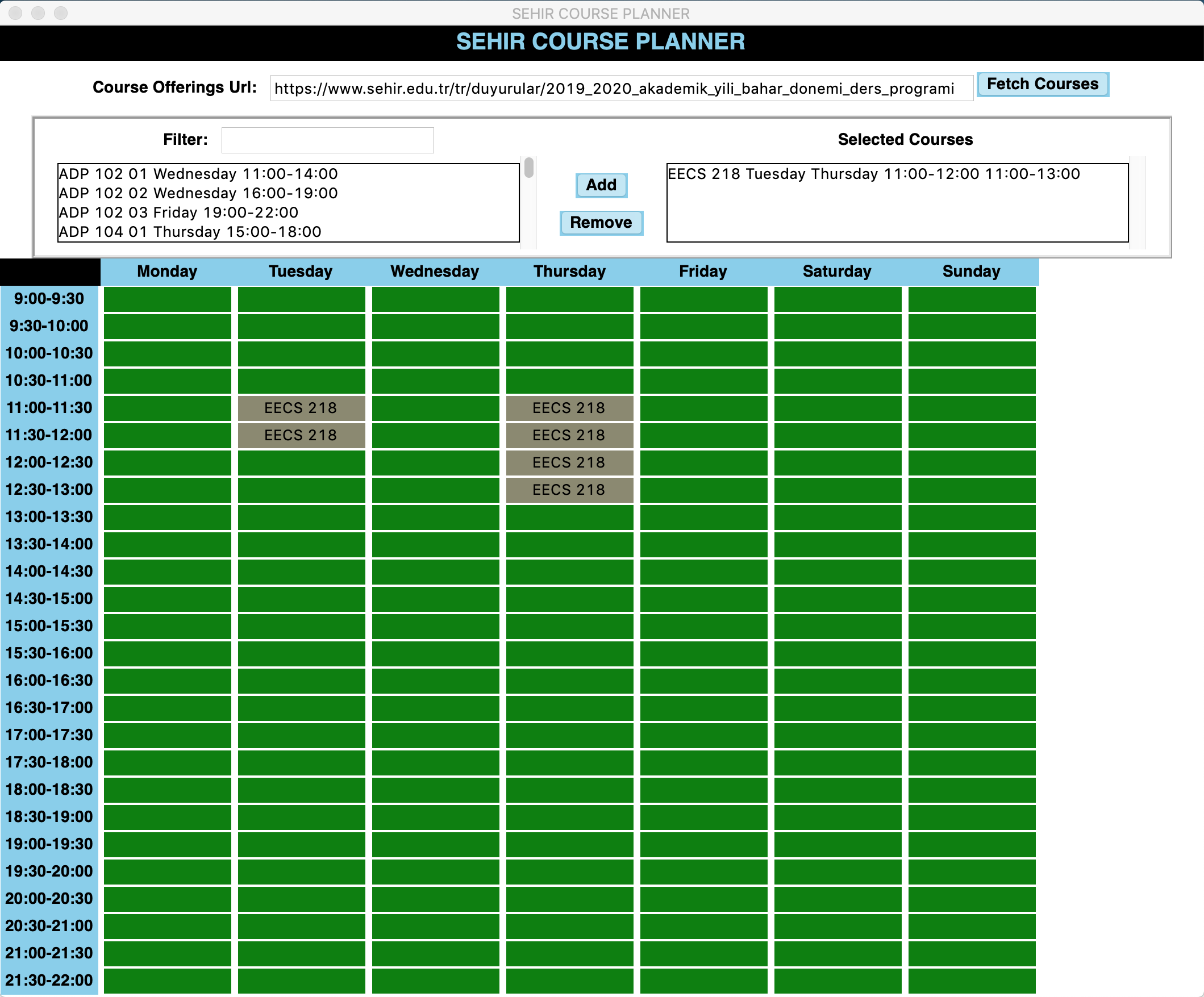
colors = [**'dodger blue'**, **'pale turquoise'**, **'dark turquoise'**, **'medium turquoise'**, **'turquoise'**, **'cyan'**, **'cadet blue'**, **'medium aquamarine'**, **'aquamarine'**, **'LemonChiffon4'**, **'cornsilk2'**,**'cornsilk4'**, **'honeydew2'**, **'dark orchid'**, **'dark violet'**, **'blue violet'**, **'purple'**]



* 1. If the selected course clashes with the existing courses, the course **WON’T** be added and a **label will be shown to the right of the time table saying “Could not be added”.**



1. Finally, a **course** or **more** could be selected from the Listbox on the right to be **removed** when the button **remove** is clicked. Needless to say, the corresponding cells should be cleaned and colored back to green.



**Implementation Notes**

* **You need to use at least 3 classes in this project. For instance, you may create classes to represent the following: GUI, Cell, Course,..etc**
* You may use Label widgets to represent every cell in the time table.
* Check this [link](https://stackoverflow.com/questions/28303346/how-to-make-a-tkinter-label-update) to be able to filter the courses in the listbox on the left whenever an entry is entered.
* To be able to update a label’s color and text, check this [link](https://stackoverflow.com/questions/28303346/how-to-make-a-tkinter-label-update).
* You may consider binding the **"<Key>"** event on the filter Entry widget to get notified whenever the user types something into this widget through the keyboard.
  + Consider this [link](https://stackoverflow.com/questions/6554805/getting-a-callback-when-a-tkinter-listbox-selection-is-changed) to be able to apply a function when a course is selected.

**Warnings:**

* You **CANNOT** use the **place** geometry manager, only **grid** and **pack** are allowed.
* Do not talk to your classmates on project topics when you are implementing your projects. Do not show or email your code to others. If you need help, talk to your TAs or the professor, not to your classmates. If somebody asks you for help, explain them the lecture slides, but do not explain any project related topic or solution. Any similarity in your source codes will have serious consequences for both parties.
* Carefully read the project document, and pay special attention to sentences that involve “should”, “should not”, “do not”, and other underlined/bold font statements.
* If you use code from a resource (web site, book, etc.), make sure that you reference those resources at the top of your source code file in the form of comments. You should give details of which part of your code is from what resource. Failing to do so may result in plagiarism investigation. Last but not the least, you need to understand code pieces that you may get some other resources. This is one of the goals of the mini projects.
* Even if you work as a group of two students, each member of the team should know every line of the code well. Hence, it is important to understand all the details in your submitted code.

**Submitting the project**

* Projects may be done individually or as a small group of two students (doing it individually is **strongly** recommended for the best learning experience). If you are doing it as a group, only **one** of the members should submit the project. File name will tell us group members (Please see the next item for file naming details).
* Submit your own code in a single Python file. Name it with your and your partner’s first and last names. As an example, if your team members are Deniz Barış and Ahmet Çalışkan, then name your code file as deniz\_baris\_ahmet\_caliskan.py (Do not use any Turkish characters in file name). If you are doing the project alone, then name it with your name and last name similar to the above naming scheme.
* Those who do not follow the above naming conventions will **get** 10% **off** of their project grade.

**Late Submission Policy:**

* **5%:** Submissions between 17:01 – 18:00 on the due date
* **10%**: Submissions between 18:01 – midnight (00:00) on the due date
* **30%**: Submissions after which are up-to 24 hours late.
* **50%**: Submissions which are up-to 48 hours late.
* Submission more than 48 hours late **will not be accepted.**

**Grading Criteria**

* **GUI design: (30)**
* **Fetching the courses and populating the listbox: (20)**
* **Coloring the available cells with yellow if the course does not clash: (10)**
* **Coloring the cells showing the clash: (15)**
* **Adding courses(along with coloring the corresponding cells): (10)**
* **Removing courses(along with uncoloring the corresponding cells): (15)**

**From your overall grade, we will deduct points by the specified percentage for the following items:**

* **Inappropriate/Cryptic variable names (-10%)**
* **Classes and objects are not used properly (-20%)**
* **Insufficient commenting (-10%).**

**Have further questions?:**

* If you need help with anything, please use the online office hours or make an appointment**.**