Matching Algorithm Use case 1

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA1 | |
| **Use Case Name** | Connect to database and get tagged faculty and subjects | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step 1:** Connect to the database and get tagged faculty and subjects |  |
| **Alternate Flow** | AtStep1: Prompt user that the system can’t connect to the database |  |
| **Precondition** | A faculty and subject is tagged | |
| **Post condition** | It was connected to the database | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 2

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA2 | |
| **Use Case Name** | Program head selects number of iterations of the algorithm | |
| **Actor(s)** | Matching Algorithm and Program head | |
| **Basic Flow** | System Response | Program head |
| **Step2:** The algorithm will now execute | **Step1:** Program head selects number of iterations |
| **Alternate Flow** | AtStep1: If Program head did not select number of iterations prompt a message |  |
| **Precondition** | Get tagged faculty and subject | |
| **Post condition** | Get retrieved faculty and subject to store to arrays | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 3

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA3 | |
| **Use Case Name** | Retrieved faculty and subject will be stored into arrays | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The algorithm will store the retrieved subjects and faculty members into subject and tagged employee array. |  |
| **Alternate Flow** |  |  |
| **Precondition** | Program head selects number of iterations | |
| **Post condition** | Check if faculty is empty | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 4

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA4 | |
| **Use Case Name** | Checking of faculty array | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The algorithm will check if the faculty array is empty, if yes a message will prompt that the no faculty is not been tagged. |  |
| **Alternate Flow** |  |  |
| **Precondition** | Storing of tagged faculty and subject into array | |
| **Post condition** | Shuffle of faculty array | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 5

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA5 | |
| **Use Case Name** | Looping of subject array to assign to subject variable | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The algorithm will get the first subject in the array and assign it to the subject variable |  |
| **Alternate Flow** |  |  |
| **Precondition** | Shuffling of faculty array | |
| **Post condition** | Matching the specialization of faculty to the subject | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 6

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA6 | |
| **Use Case Name** | Matching the Speciation of the faculty to the subject | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The algorithm get all the faculty has the specialization of the subject. |  |
| **Alternate Flow** |  |  |
| **Precondition** | Loop of subject array | |
| **Post condition** | Loop of faculty array | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 7

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA7 | |
| **Use Case Name** | Loop of the faculty members | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The matching algorithm will loop the faculty array to be assigned to the faculty variable |  |
| **Alternate Flow** |  |  |
| **Precondition** | Getting all the faculty that has the specialization of the subject | |
| **Post condition** | Check if faculty has the specialization of the subject | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 8

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA8 | |
| **Use Case Name** | Check if the faculty has the specialization of subject. | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The algorithm will check If the faculty specialization is equal to the subject specialization then proceed. |  |
| **Alternate Flow** |  |  |
| **Precondition** | Loop of faculty members | |
| **Post condition** | Check if faculty has the specialization of the subject | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 9

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA9 | |
| **Use Case Name** | Add 60 percent to the faculty loading score | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The result of the matched subject with correlating to the faculty the algorithm will add 60 percent to the faculty loading score. |  |
| **Alternate Flow** |  |  |
| **Precondition** | Check if the faculty has the specialization of subject. | |
| **Post condition** | Check faculty type. | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 10

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA10 | |
| **Use Case Name** | The algorithm will check the Faculty type | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The algorithm will check if the faculty’s availability is full time, if yes proceed to checking of conflicts. |  |
| **Alternate Flow** | At Step2 : check faculty availability and match it to the subject sched |  |
| **Precondition** | Check faculty type | |
| **Post condition** | Faculty availability match to the subject schedule | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 11

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA11 | |
| **Use Case Name** | Part time faculty members | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The algorithm will check the faculty availability and match it to the subject schedule, if yes proceed to checking of conflicts |  |
| **Alternate Flow** | At Step2: Get another faculty from the faculty array |  |
| **Precondition** | Check if faculty is full time | |
| **Post condition** | Check conflicts on rules | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 12

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA12 | |
| **Use Case Name** | Checking of Rules | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The algorithm will proceed to the checking of rules.  **Step2:** The algorithm will check the 1st Rule in which the function is to limit the consecutive class of the faculty member by upto 2 classes only.  **Step 3:** The algorithm will check the 2nd Rule in which the function is to limit the units of the full-time faculty members by upto 18 units only and part time faculty members by upto 15 units only. |  |
| **Alternate Flow** | At Step2: The system will raise flag to the occupied schedules of the faculty member. If the faculty member already has 2 consecutive classes, the system will also raise flag to the next schedule of the faculty member and will go back to the Check on conflicts rules process.  At Step 3: The system will output a message prompt  “Maximum number of subjects have been reached” if the maximum number of units for each type of faculty member have already reached. |  |
| **Precondition** | Check faculty availability | |
| **Post condition** | Check | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 13

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA13 | |
| **Use Case Name** | Selecting OTE (Online Teacher Evaluation) Score of the faculty member | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The algorithm will select the OTE Score of the faculty member from the Faculty Load Database.  **Step2:** The algorithm will multiply the OTE Score to 10%  **Step3:** The algorithm will store the OTE Result to the Percent Score table. |  |
| **Alternate Flow** |  |  |
| **Precondition** | Retrieved the OTE Score from HR Database | |
| **Post condition** | Select HoT (History of Teachings) Score of the faculty member. | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 14

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA14 | |
| **Use Case Name** | Selecting HoT (History of Teachings) Score of the faculty member | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The algorithm will select the HoT Score of the faculty member from the Faculty Load Database.  **Step2:** The algorithm will multiply the HoT Score to 30%  **Step3:** The algorithm will store the HoT Result to the Percent Score table. |  |
| **Alternate Flow** |  |  |
| **Precondition** | Retrieved the HoT Score from Registrar Database | |
| **Post condition** | Select HoT (History of Teachings) Score of the faculty member. | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 15

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA15 | |
| **Use Case Name** | Getting the total faculty percentage | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The algorithm will compute for the final faculty percentage |  |
| **Alternate Flow** |  |  |
| **Precondition** | Getting different scores from the Percent Score table. | |
| **Post condition** | Faculty array loop | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |

Matching Algorithm Use case 16

|  |  |  |
| --- | --- | --- |
| **Use Case Number** | MA16 | |
| **Use Case Name** | Selecting the faculty with the highest percentage | |
| **Actor(s)** | Matching Algorithm | |
| **Basic Flow** | System Response |  |
| **Step1:** The algorithm will compare the total final percentage of each faculty  **Step2:** The algorithm will choose the faculty member with the highest percentage. |  |
| **Alternate Flow** | At Step 2:  If there is no other faculty to compare, the algorithm will skip this step and will assign the faculty for that certain subject right away. |  |
| **Precondition** | Selected the final percentage of different faculty members. | |
| **Post condition** | Assigned faculty will be stored to the Faculty Load Database. | |
| **Special Requirements** | The data that will be supplied for the values must be correct. | |