CS 6235 Real-Time Systems, Fall 2016

Course Recommender

Final Project Report

Team Members:

- Chaitanya R. Maniar
- Ishika Roy
- Sampada Upasani

Motivation -

Usually universities providing higher education have a high ratio of enrolment of students who may be unfamiliar with the subtleties of navigating their way through the degree program offered. Although the students are guided by an advisor each semester, the student still needs a thorough individual evaluation based on his own interest and skills as well as information of the ongoing general trend. The ongoing general trend can be observed from the courses previous students had chosen. To ease a new student through this course selection process and to give him a fair idea of all appropriate and available options, we decided to create Course Recommender, which is an application to help students make advantageous and informed choices about their education.

The traditional approach to this problem would be to either base the decision on a word to word referral of courses by peers or to seek the advice of a guidance counselor. Both of these approaches are quite time consuming and might not be absolutely accurate as they are based on individual perspective.

Our proposed system recommends courses based on their popularity and also the student’s technical area of interest or specialization. Furthermore, it takes into account the term the student is currently in, that is, fall or spring, and recommends the courses offered in those semesters. The recommendation also weighs in the data from other students who had previously taken the same course and accordingly recommends the subsequent courses chosen.

Related Work -

Degree Compass -

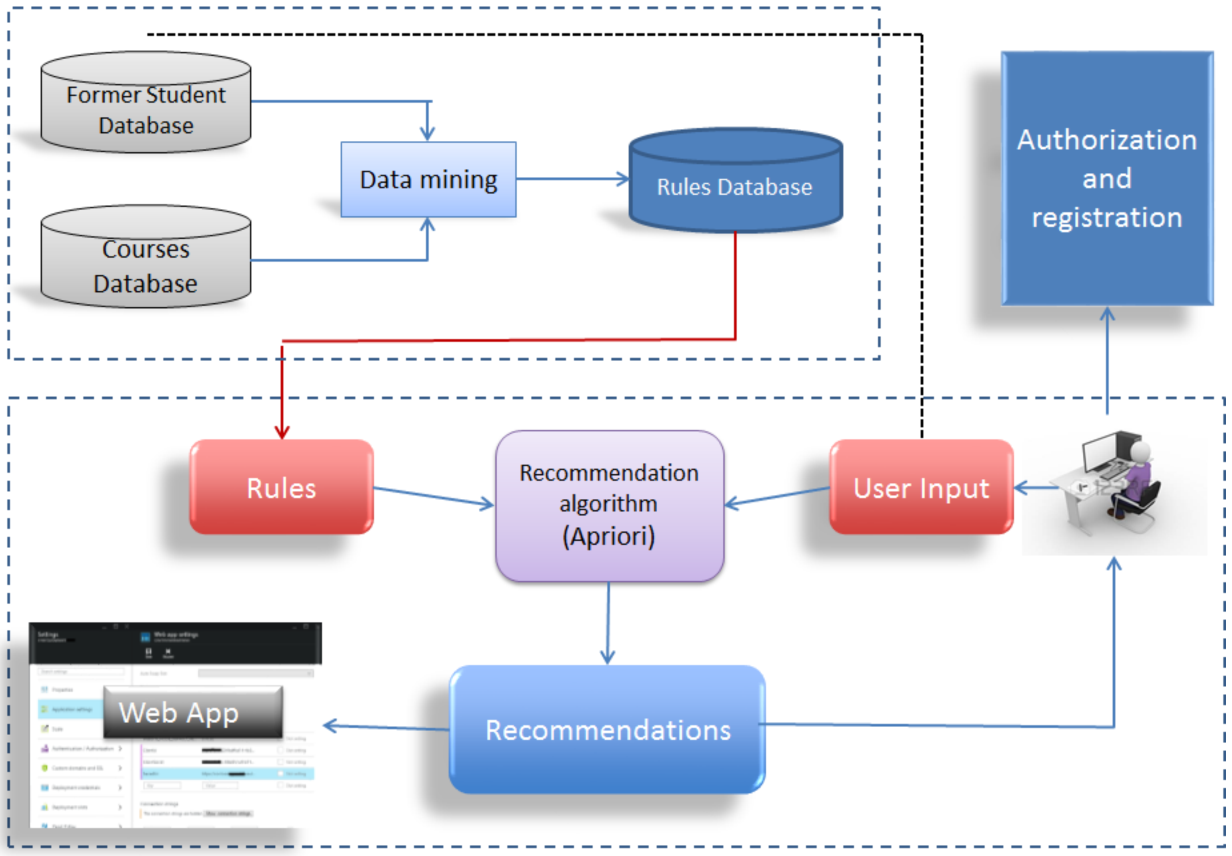
Degree Compass is inspired by recommendation systems implemented by companies such as Netflix, Amazon, and Pandora. It successfully pairs current students with the courses that best fit their talents and program of study for upcoming semesters. The model combines past students' grades with each particular student's transcript to make individualized recommendations.
Predictive analytics techniques based on grade and enrolment data is used to rank courses. From the courses that apply directly to the student's program of study, the system selects those the ones that fit best with the course sequence in their degree and are the most central to the university curriculum as whole. That ranking is then overlaid with a model that predicts the courses in which each student will achieve the best grades. Additionally, the system most strongly recommends a course that is necessary for a student to graduate, that is core to the university curriculum and the student's major, and that the student is expected to succeed in academically.

Student Course Recommender (SCR) -

It is a learning system that can estimate the students' interests in the courses given information such as type of education, major and courses previously taken. Learning is achieved by using old students' course choices as training examples. SCR tries to recommend courses with highest probability with help of Bayesian theory by evaluating maximum likelihood. In the Bayesian model, the variables represent courses and education with each variable having states of true or false. The SCR has a unique user model for a user that consists of all the courses taken in previous years by that student. The system also has a general model which stores the probabilities of taking a course based on other users' course choices.

A static user model is created at the beginning of the session and it does not change to avoid conflicts in case new data entered is inconsistent. The user interface provides links for each institution at the university. When the student gives information, all courses already taken, or the ones the student doesn't have prerequisite for and the courses that overlaps with the courses taken are removed. The user interacts with the system using GUI implemented in Java.

System Architecture -



The project consists of two phases:

Offline phase -

Database:

- There are two databases of former students, one which comprises of the courses they have taken in the spring term and one which comprises of courses taken in the fall term, as the courses offered in both terms are different. Additionally, there are databases consisting of all courses listed according to TIG (Technical Interest Group) and Specialization offered by the CS as well as ECE department.

Data Mining:

- The data from these databases are analyzed and filtered to obtain a transaction format which can then be used as input for the recommendation algorithm. For example, in the former student's database, we are not concerned with the general student information or knowing which student has taken which courses. Rather, we only need to have the groups of courses taken by a student in a particular semester to generate the association rules.

Association rules:

- Once we have filtered the data, we can apply the recommendation algorithm to generate association rules. The algorithm we are using is the Apriori algorithm (explained later) which analyzes the data and generates a set of rules that provides us the association from one course to another for all the courses in the refined database of former students.
- All rules are generated along with support, confidence and lift, and stored in a rules database for further use.

Online phase -

Registration:

- Initially, the user has to register into the system with a username and password. At the registration page, the user needs to enter his personal information i.e. name, department, TIG/Specialization, current term (fall or spring) and the previous semester courses they had opted for, provided that it is not their first semester.
- The courses obtained are added to the former student database and other credentials are saved in a registration database along with username and password.

Login:

- If the user logs in again, he is authenticated using the registration database and his details are pulled up and displayed on the page.
- After logging in, the user has two options - he can either select his TIG to get all courses recommended for the particular TIG or he can select the courses he plans to take for which the recommender generates recommended courses that can be taken along with it, according to the rules in the rules database.

Recommendation:

- The Apriori algorithm takes the user input of either TIG and courses and searches the rules with some pre-set thresholds for support, confidence and lift, and consequently returns the corresponding courses as recommendations.

The Apriori Algorithm -

Concept:

The Apriori Algorithm is an influential algorithm for mining frequent itemsets for generating association rules. It uses a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as candidate generation), and groups of candidates are tested against the data. Apriori is designed to operate on database containing transactions (for example, collections of items bought by customers, or details of a website frequentation).

It follows the principle that a subset of a frequent itemset must also be a frequent itemset i.e., if $\{A, B\}$ is a frequent itemset, both $\{A\}$ and $\{B\}$ should be frequent itemsets. These frequent itemsets are then used to generate association rules.
The association rules are generated in accordance to the transactions' support, confidence and lift.

Association Rules:

An association rule has two parts, an antecedent (if) and a consequent (then). An antecedent is an item found in the data. A consequent is an item that is found in combination with the antecedent. Association rules are created by analyzing data for frequent if/then patterns and using the criteria support and confidence to identify the most important relationships. Association rules are then required to satisfy a specified minimum support and a specified minimum confidence. Association rule generation is usually split up into two separate steps:

1) A minimum support threshold is applied to find all frequent item sets in a database.
2) A minimum confidence constraint is applied to these frequent item-sets in order to form rules.

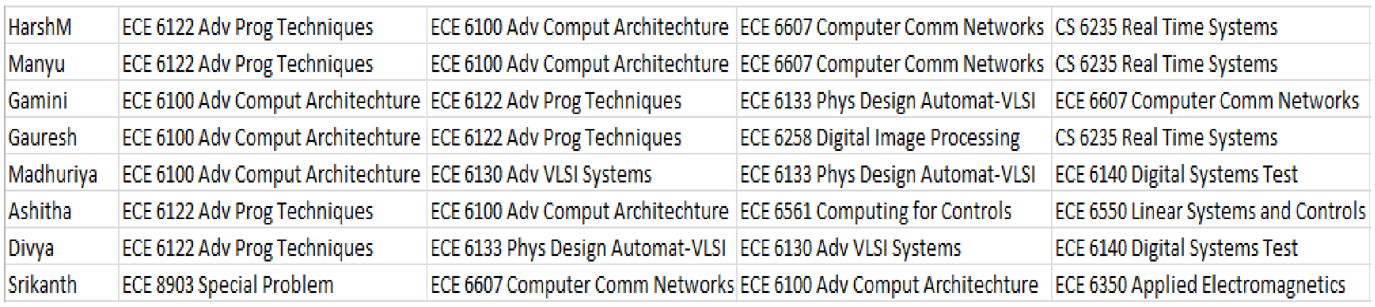
We have chosen support as 0.11 (11%) and confidence as 0.40 (40%) which is chosen in accordance with amount of data that was acquired. The ideal confidence threshold would be between 60-75% based on the application, and the size of the dataset.

Support is an indication of how frequently the items appear in the database. Confidence indicates the number of times the if/then statements have been found to be true. The confidence value of a rule, $X \Rightarrow Y$, with respect to a set of transactions T, is the proportion of the transactions that contains X which also contains Y.

The lift of a rule is defined as the ratio of the observed support to that expected if X and Y were independent. If the rule had a lift of 1, it would imply that the probability of occurrence of the antecedent and that of the consequent are independent of each other. When two events are independent of each other, no rule can be drawn involving those two events. If the lift is > 1, that lets us know the degree to which those two occurrences are dependent on one another, and makes those rules potentially useful for predicting the consequent in future data sets. The value of lift is that it considers both the confidence of the rule and the overall data set.

$$Support = \frac{frq(X,Y)}{N}, \qquad Confidence = \frac{frq(X,Y)}{frq(X)}, \qquad Lift = \frac{Support}{Supp(X) \times Supp(Y)}$$

Example:



HarshM	ECE 6122 Adv Prog Techniques	ECE 6100 Adv Comput Architechture	ECE 6607 Computer Comm Networks	CS 6235 Real Time Systems
Manyu	ECE 6122 Adv Prog Techniques	ECE 6100 Adv Comput Architechture	ECE 6607 Computer Comm Networks	CS 6235 Real Time Systems
Gamini	ECE 6100 Adv Comput Architechture	ECE 6122 Adv Prog Techniques	ECE 6133 Phys Design Automat-VLSI	ECE 6607 Computer Comm Networks
Gauresh	ECE 6100 Adv Comput Architechture	ECE 6122 Adv Prog Techniques	ECE 6258 Digital Image Processing	CS 6235 Real Time Systems
Madhuriya	ECE 6100 Adv Comput Architechture	ECE 6130 Adv VLSI Systems	ECE 6133 Phys Design Automat-VLSI	ECE 6140 Digital Systems Test
Ashitha	ECE 6122 Adv Prog Techniques	ECE 6100 Adv Comput Architechture	ECE 6561 Computing for Controls	ECE 6550 Linear Systems and Controls
Divya	ECE 6122 Adv Prog Techniques	ECE 6133 Phys Design Automat-VLSI	ECE 6130 Adv VLSI Systems	ECE 6140 Digital Systems Test
Srikanth	ECE 8903 Special Problem	ECE 6607 Computer Comm Networks	ECE 6100 Adv Comput Architechture	ECE 6350 Applied Electromagnetics

In the snippet of the former student database for fall term, we are provided with the students and the four courses that they had chosen. The rule -

$$ECE\ 6122\ Adv\ Prog\ Techniques\ \Rightarrow ECE\ 6100\ Adv\ Comp\ Arch$$

Is generated with

$Support = \frac{6}{8} = 0.75$ $\quad$ $Confidence = \frac{5}{6} = 0.83$ $\quad$ $Lift = \frac{6/8}{6/8 \times 7/8} = 1.142$

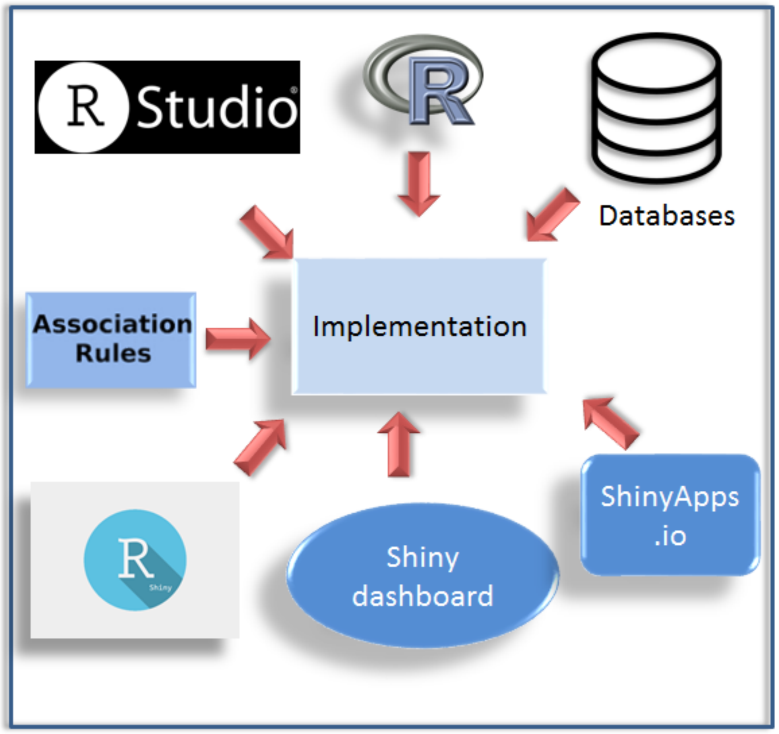
Algorithm:

Apriori uses breadth-first search and a Hash tree structure to count candidate item sets efficiently. It generates candidate item sets of length $\boldsymbol{K}$ from item sets of length $\boldsymbol{K-1}$. Then it prunes the candidates which have an infrequent sub pattern. According to the downward closure lemma, the candidate set contains all frequent $\boldsymbol{K}$-length item sets. After that, it scans the transaction database to determine frequent item sets among the candidates.

Apriori Pseudo Code -

```
Apriori($T, \epsilon$)
    $L_1 \leftarrow \{\text{large } 1 - \text{itemsets}\}$
    $k \leftarrow 2$
    while $L_{k-1} \neq \emptyset$
        $C_k \leftarrow \{a \cup \{b\} \mid a \in L_{k-1} \wedge b \notin a\} - \{c \mid \{s \mid s \subseteq c \wedge |s| = k-1\} \nsubseteq L_{k-1}\}$
        for transactions $t \in T$
            $C_t \leftarrow \{c \mid c \in C_k \wedge c \subseteq t\}$
            for candidates $c \in C_t$
                $count[c] \leftarrow count[c] + 1$
        $L_k \leftarrow \{c \mid c \in C_k \wedge\ count[c] \geq \epsilon\}$
        $k \leftarrow k + 1$
    return $\bigcup_k L_k$
```

Implementation -



R Packages:

- Shiny
- Shinydashboard
- Arules
- stringr
- datasets

Tools Used -

R Language:

R is a programming language and software environment for statistical computing and graphics supported by the R Foundation for Statistical Computing. The R language is widely used among statisticians and data miners for developing statistical software and data analysis. Many of R's standard functions are written in R itself, which makes it easy for users to follow the algorithmic choices made. For computationally intensive tasks, C, C++, and Fortran code can be linked and called at run time. Advanced users can write C, C++, Java, .NET or Python code to manipulate R objects directly.

We use **Rstudio** which integrates with R as an IDE (Integrated Development Environment) to provide further functionality. RStudio combines a source code editor, build automation tools and a debugger. The program is run locally as a regular desktop application.

Packages:

The capabilities of R are extended through user-created packages, which allow specialized statistical techniques, graphical devices (such as the ggplot2 package), import/export capabilities, reporting tools (knitr, Sweave), etc. These packages are developed primarily in R, and sometimes in Java, C, C++, and Fortran.

The packages used are:

- arules
- Stringr
- Shiny
- ShinyDashboard

The arules package: Provides the infrastructure for representing, manipulating and analyzing transaction data and patterns (frequent itemsets and association rules). Also provides interfaces to C implementations of the association mining algorithms Apriori and Eclat.

The stringr package: Allows for fast, correct, consistent, portable, as well as convenient character string/text processing in every locale and any native encoding. Among available features there are: pattern searching, random string generation, case mapping, string transliteration, concatenation, etc. We require this to search our databases and rules for user-entered courses so as to suggest the corresponding courses that surpass the thresholds for support, confidence and lift.

The Shiny package: Makes it incredibly easy to build interactive web applications with R. Automatic "reactive" binding between inputs and outputs and extensive pre-built widgets make it possible to build beautiful, responsive, and powerful applications.

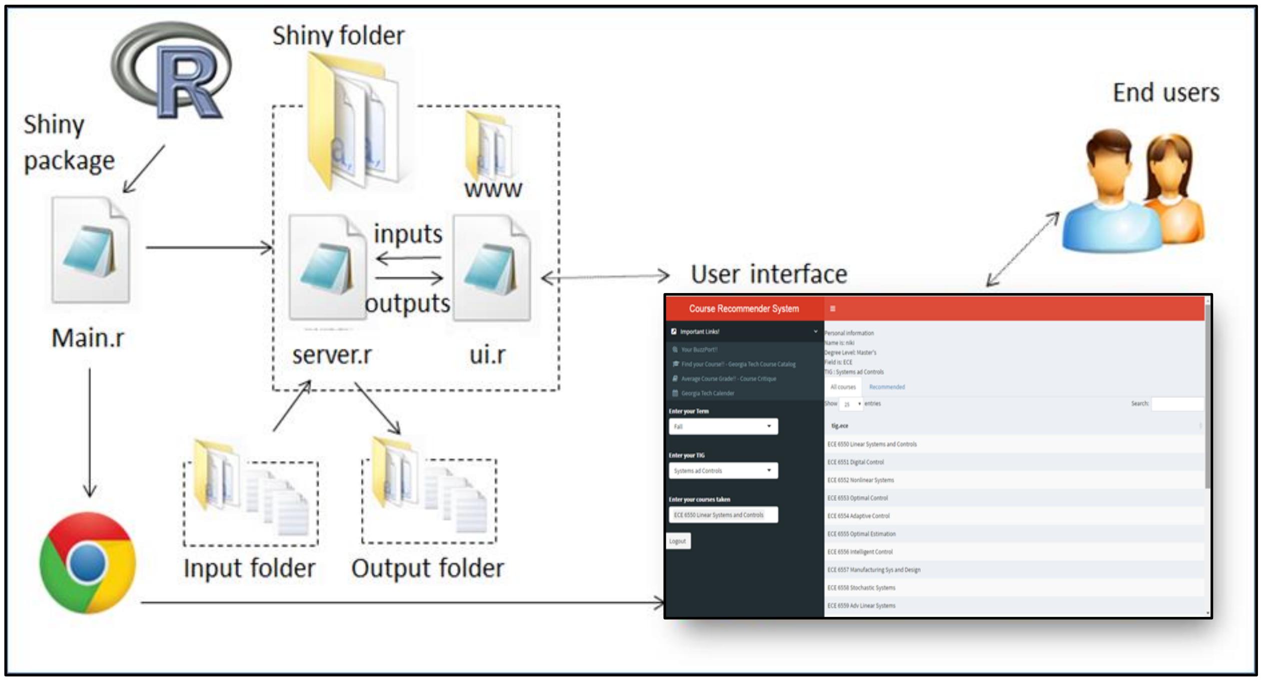
The ShinyDashboard package: Creates dashboards with 'Shiny'. This package provides a theme on top of 'Shiny', making it easy to create attractive dashboards.

R Shiny framework

Shiny is web application framework for R. It allows customization of the application's user interface to provide an elegant environment for displaying user-input controls and simulation output–where the latter simultaneously updates with changing input. Unlike other Web-page design methods, only previous experience with the R programming language is required. R and Shiny maintain Berkeley Madonna's key feature of "reactively" updating output in response to changing input by means of widgets but owing to the flexible R language and combination with extension packages. We believe such Shiny Web applications have the potential to provide a convenient method for creating interactive web applications.

Using R Shiny:

Shiny is a package for R developed by RStudio, and needs to be installed in a given R installation. Shiny applications are built using two R scripts that communicate with each other: a user-interface script (ui.R), which controls layout and appearance; and a server script (server.R), incorporating instructions for user-input, processing data, and output by utilizing the R language and functions from user installed packages. The ui.R script encodes instructions for the application's layout, appearance, user-input widgets, and the output to be displayed. The main elements describing the user-interface of this application are shown below:



Personal information
Name is: niki
Degree Level: Master's
Field is: ECE
TIG : Systems ad Controls

Enter your Term: Fall
Enter your TIG: Systems ad Controls
Enter your courses taken: ECE 6550 Linear Systems and Controls

Show 25 entries Search:

tig.ece
ECE 6550 Linear Systems and Controls
ECE 6551 Digital Control
ECE 6552 Nonlinear Systems
ECE 6553 Optimal Control
ECE 6554 Adaptive Control
ECE 6555 Optimal Estimation
ECE 6556 Intelligent Control
ECE 6557 Manufacturing Sys and Design
ECE 6558 Stochastic Systems
ECE 6559 Adv Linear Systems

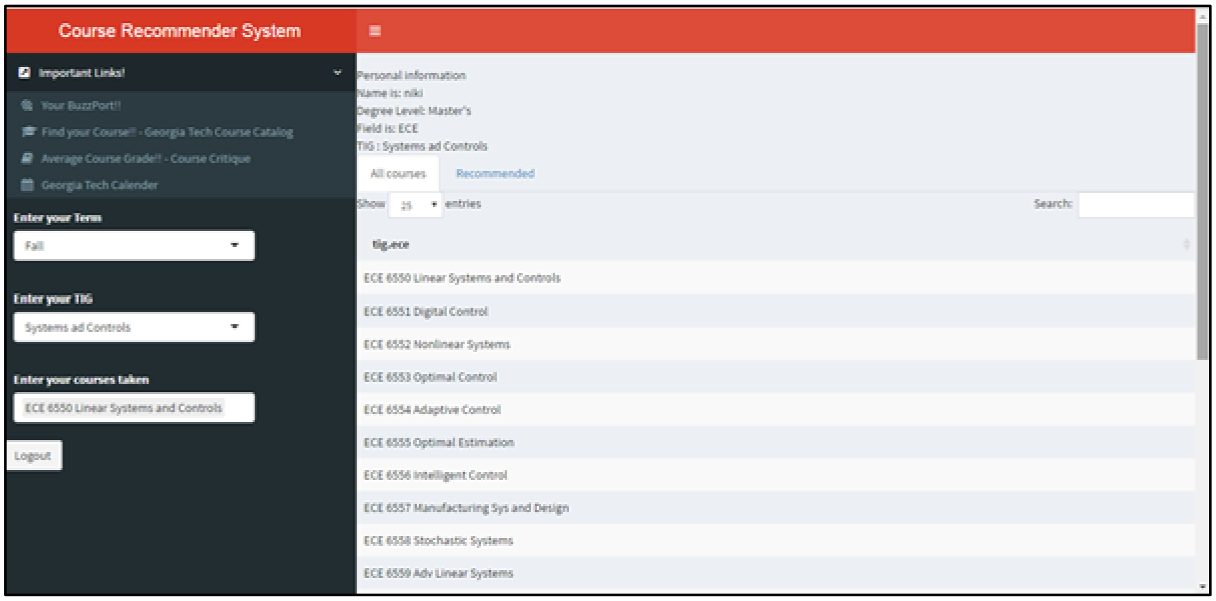
Building a User-Interface (ui.R):

RShiny Dashboard:

Shiny provides us with the package ‘shinydashboard’ which makes it easy to create user friendly interfaces. All code for the contents of the user-interface is required to be within the brackets of a layout function. Layout functions such as dashboardPage (there are a number available) create a canvas for the interface and use fluidRow to position user-input widgets (textInput—a function for creating a textbox) and output (textOutput—function for displaying the output). Each layout function has its own framework for positioning elements. However, they all follow the same hierarchical structure where element functions (such as widgets— selectInput) are embraced within a positioning function (such as fluidRow), within a layout function (shinydashboardPage). The basic framework of a dashboard looks like below:

```
library(shinydashboard)
dashboardPage(
 dashboardHeader(),
 dashboardSidebar(),
 dashboardBody()
```

User Interface (UI) using Shiny dashboard for this application:



Course Recommender System

Important Links!
- Your BuzzPort!!
- Find your Course!! - Georgia Tech Course Catalog
- Average Course Grade!! - Course Critique
- Georgia Tech Calender

Enter your Term: Fall

Enter your TIG: Systems ad Controls

Enter your courses taken: ECE 6550 Linear Systems and Controls

Logout

Personal information
Name is: niki
Degree Level: Master's
Field is: ECE
TIG : Systems ad Controls

All courses | Recommended

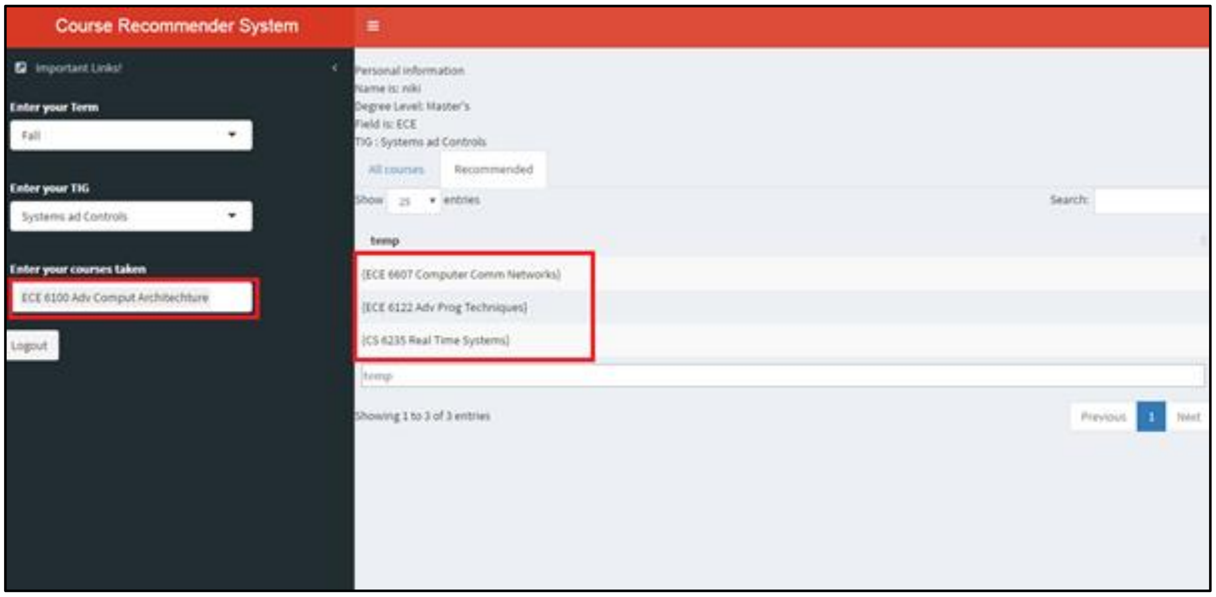
Show 25 entries | Search:

tig.ece
ECE 6550 Linear Systems and Controls
ECE 6551 Digital Control
ECE 6552 Nonlinear Systems
ECE 6553 Optimal Control
ECE 6554 Adaptive Control
ECE 6555 Optimal Estimation
ECE 6556 Intelligent Control
ECE 6557 Manufacturing Sys and Design
ECE 6558 Stochastic Systems
ECE 6559 Adv Linear Systems

Shiny calls elements in the ui.R script sequentially, i.e., from top to bottom, and left to right. Within a layout function, elements inside are ordered in the sequence they are to appear in the user-interface. Error messages related to Shiny (and other loaded R packages) appear in the loaded Web browser upon application initiation, and once the application is terminated they also appear in the R Console.

Server.R:

The structure of server.R code plays an important role in defining instructions for the application, while minimizing redundant computation and maximizing application speed. The function, shinyServer, requires an input object (values called from ui.R) and an output object (objects to be called by ui.R). Objects that change depending on input from widgets in the ui.R, such as input$Course are termed as “reactive”. Every time a user inputs a new course, the value for the reactive object updates to reflect this change. Render functions are used for defining and processing reactive objects for (ex: renderUI - uiOutput, renderDatatable – dataTableOutput). An example of this reactivity is shown below.



Enter your Term: Fall

Enter your TIG: Systems ad Controls

Enter your courses taken: ECE 6100 Adv Comput Architechture

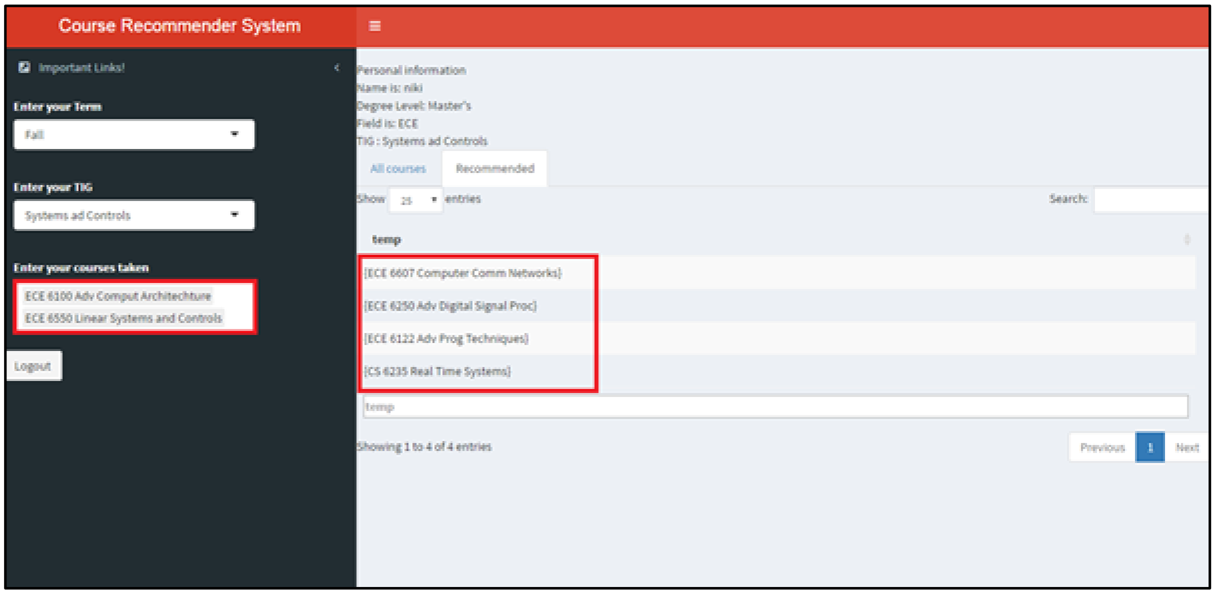
Personal information
Name is: niki
Degree Level: Master's
Field is: ECE
TIG : Systems ad Controls

Show 25 entries   Search:

temp
[ECE 6607 Computer Comm Networks]
[ECE 6122 Adv Prog Techniques]
[CS 6235 Real Time Systems]

Showing 1 to 3 of 3 entries

RenderDataTable displays the highlighted recommended course when user selects course as ‘ECE6100 Adv Computer Architechture’



Enter your Term: Fall

Enter your TIG: Systems ad Controls

Enter your courses taken:
ECE 6100 Adv Comput Architechture
ECE 6550 Linear Systems and Controls

Personal information
Name is: niki
Degree Level: Master's
Field is: ECE
TIG : Systems ad Controls

Show 25 entries   Search:

temp
[ECE 6607 Computer Comm Networks]
[ECE 6250 Adv Digital Signal Proc]
[ECE 6122 Adv Prog Techniques]
[CS 6235 Real Time Systems]

Showing 1 to 4 of 4 entries

RenderDataTable displays the highlighted recommended courses as soon as the user changes its course selection to 'ECE6100 Adv Computer Architecture' and 'ECE6550 Linear Systems and Controls'.

Shinyapps.io:

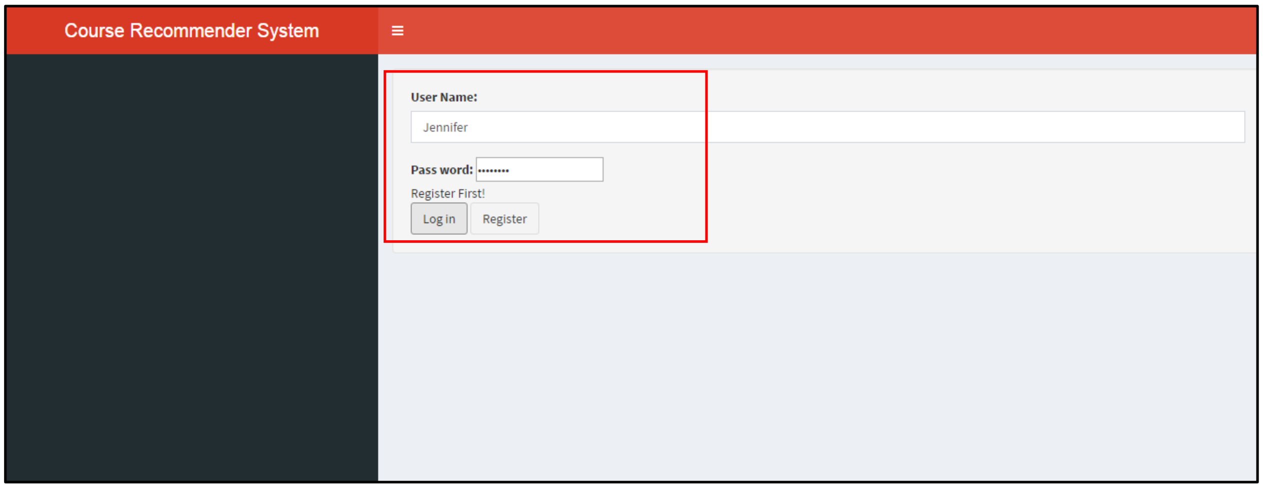
Shinyapps.io is a platform as a service (PaaS) for hosting Shiny web apps (applications). The service runs in the cloud on shared servers that are operated by RStudio. Shinyapps.io is secure-by-design. Each Shiny application runs in its own protected environment and access is always SSL encrypted.

Web Application –

https://rts-course-recommender.shinyapps.io/course_recommender_system/

Registration:

New user tries to log in, user is asked to register first:



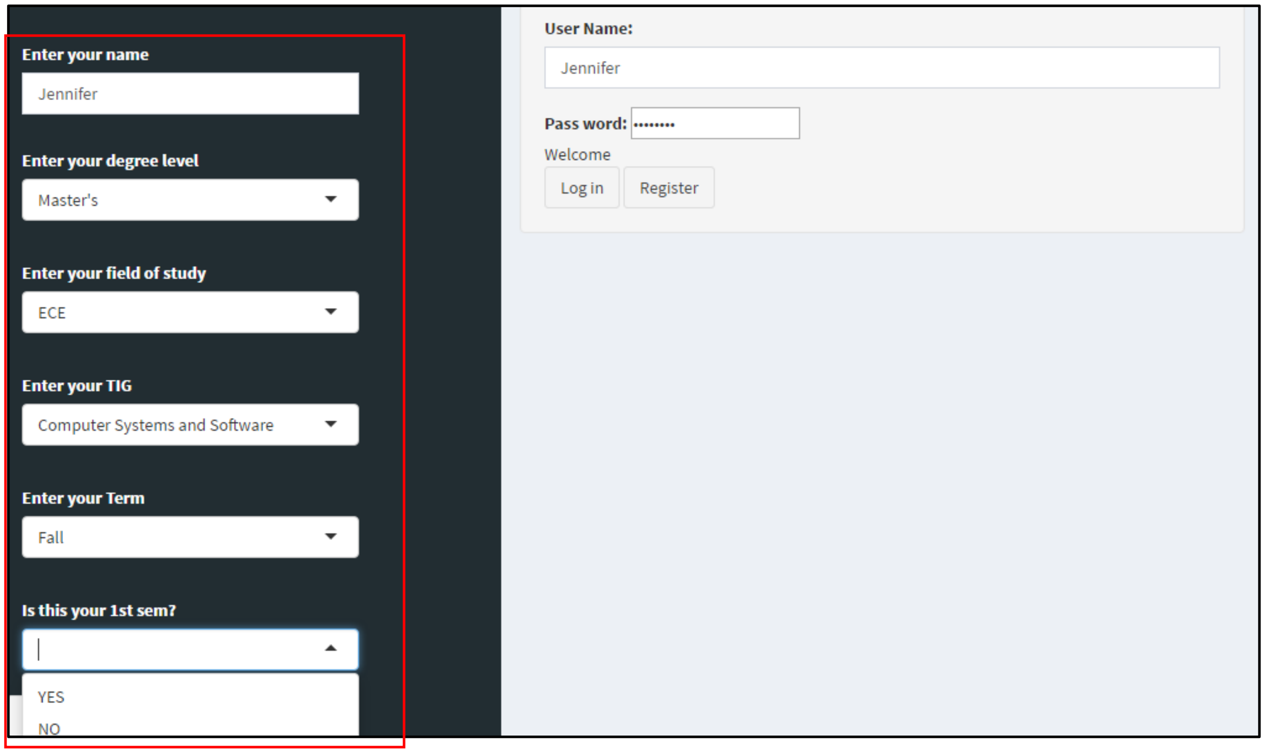
Course Recommender System

User Name: Jennifer

Pass word: ••••••••

Register First!

After clicking register, fill in personal information:



Enter your name: Jennifer

Enter your degree level: Master's

Enter your field of study: ECE

Enter your TIG: Computer Systems and Software

Enter your Term: Fall

Is this your 1st sem?

YES

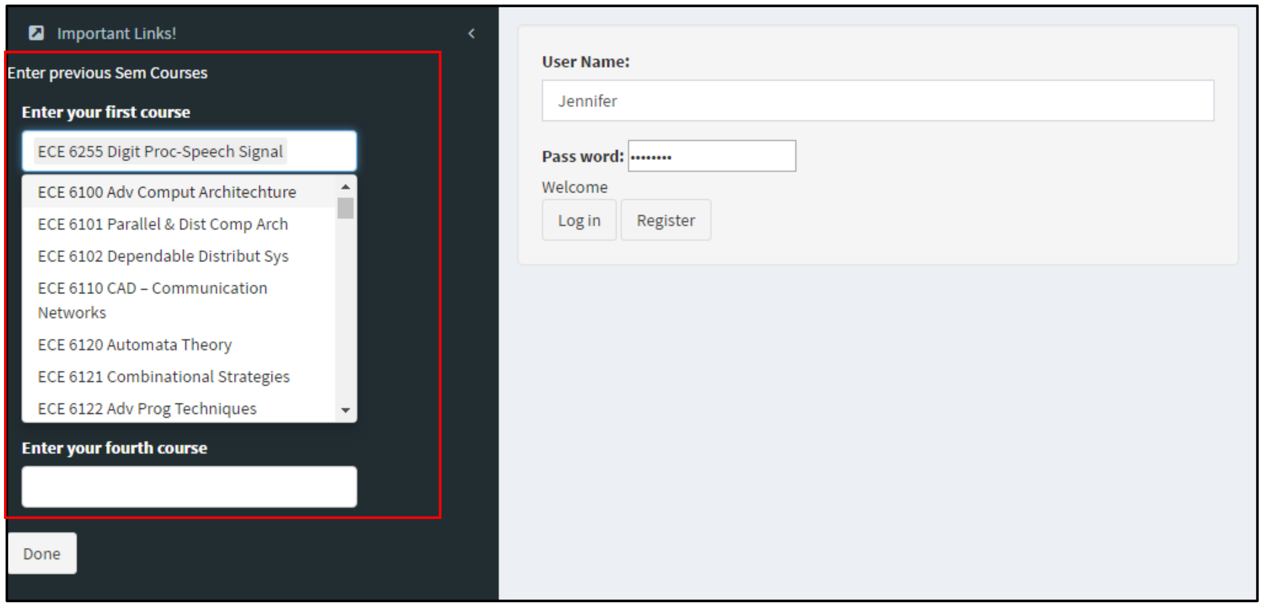
NO

User Name: Jennifer

Pass word: ••••••••

Welcome

If the user has taken courses in his/her previous semesters, he is required to enter them.



Enter previous Sem Courses

Enter your first course

ECE 6255 Digit Proc-Speech Signal

ECE 6100 Adv Comput Architechture
ECE 6101 Parallel & Dist Comp Arch
ECE 6102 Dependable Distribut Sys
ECE 6110 CAD – Communication Networks
ECE 6120 Automata Theory
ECE 6121 Combinational Strategies
ECE 6122 Adv Prog Techniques

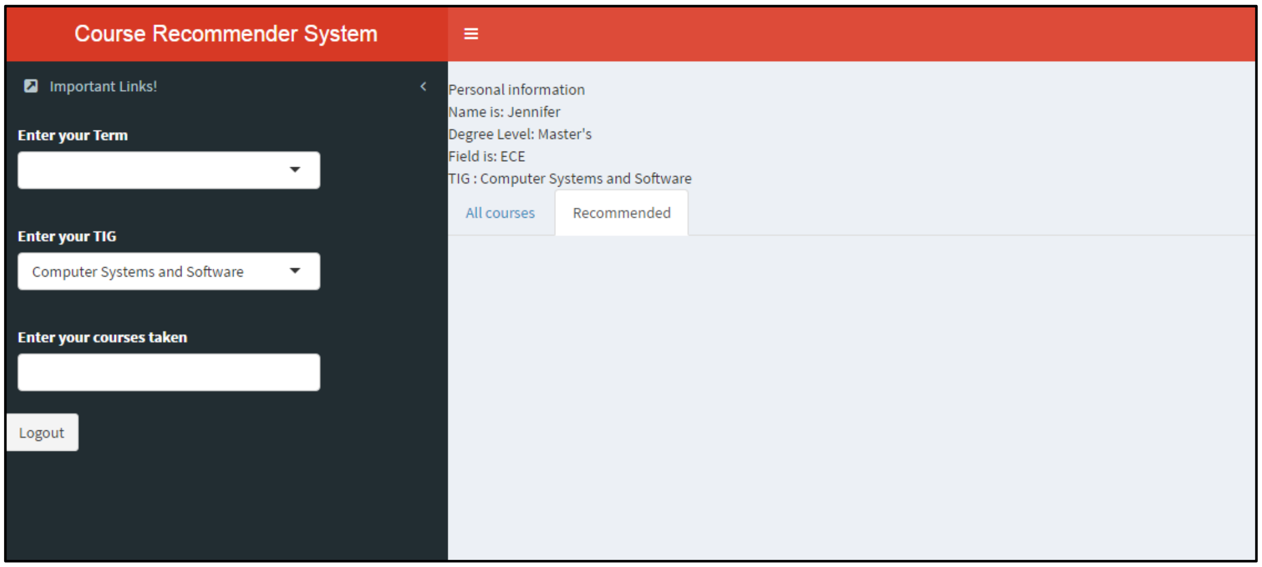
Enter your fourth course

User Name: Jennifer

Pass word: ••••••••

Welcome

Login:



Enter your Term

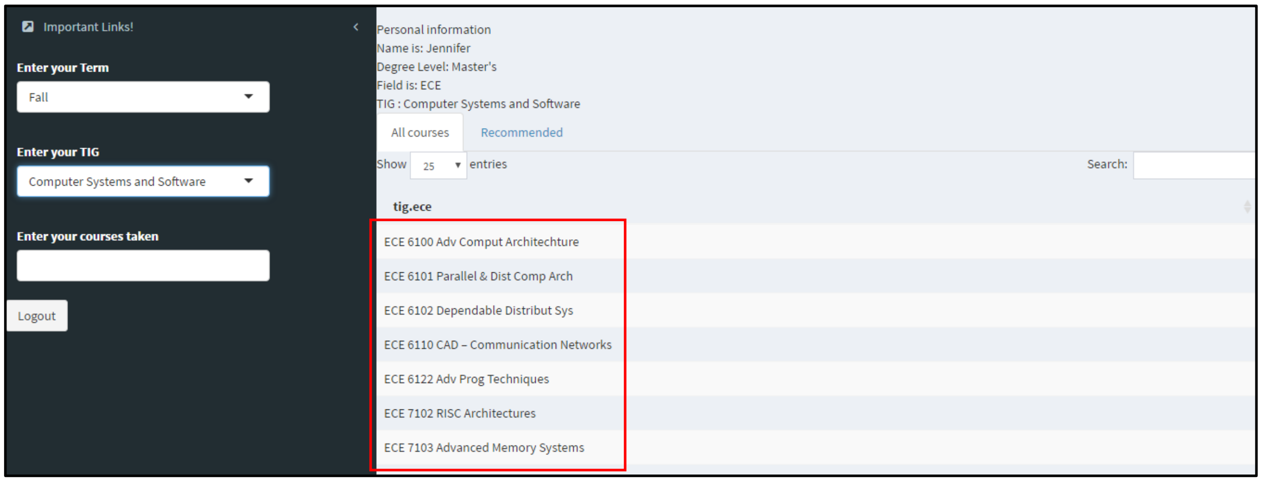
Enter your TIG: Computer Systems and Software

Enter your courses taken

Personal information
Name is: Jennifer
Degree Level: Master's
Field is: ECE
TIG : Computer Systems and Software

Recommendation:

All courses offered for specialization chosen:



Enter your Term: Fall

Enter your TIG: Computer Systems and Software

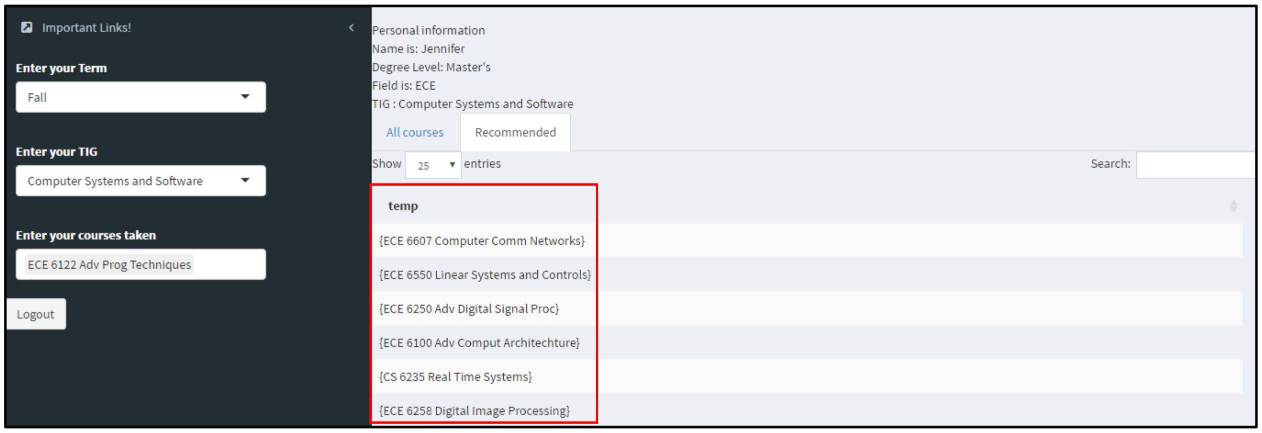
Enter your courses taken:

Personal information
Name is: Jennifer
Degree Level: Master's
Field is: ECE
TIG : Computer Systems and Software

Show 25 entries   Search:

tig.ece
ECE 6100 Adv Comput Architechture
ECE 6101 Parallel & Dist Comp Arch
ECE 6102 Dependable Distribut Sys
ECE 6110 CAD – Communication Networks
ECE 6122 Adv Prog Techniques
ECE 7102 RISC Architectures
ECE 7103 Advanced Memory Systems

Recommended courses for courses chosen:



Enter your Term: Fall

Enter your TIG: Computer Systems and Software

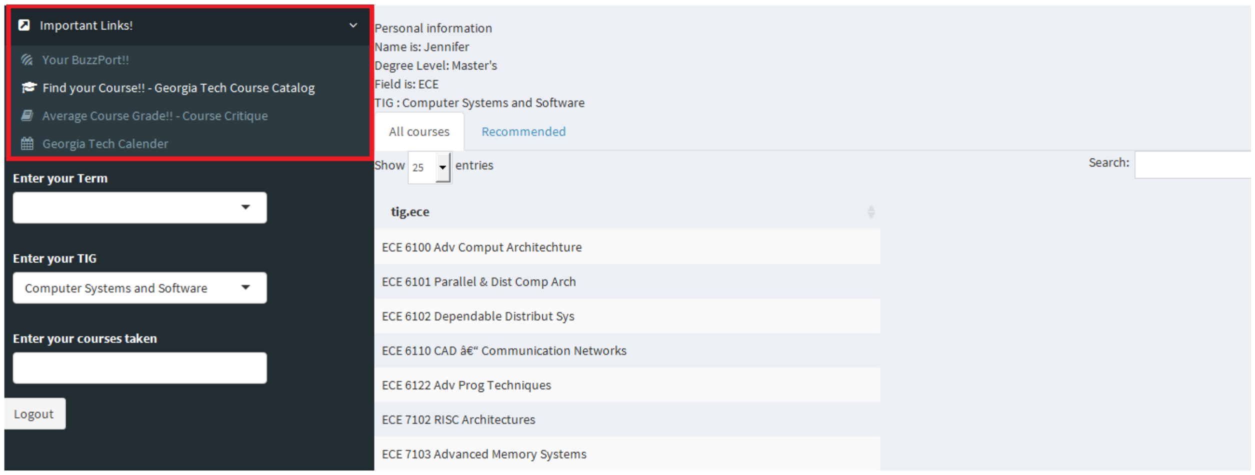
Enter your courses taken: ECE 6122 Adv Prog Techniques

Personal information
Name is: Jennifer
Degree Level: Master's
Field is: ECE
TIG : Computer Systems and Software

Show 25 entries   Search:

temp
{ECE 6607 Computer Comm Networks}
{ECE 6550 Linear Systems and Controls}
{ECE 6250 Adv Digital Signal Proc}
{ECE 6100 Adv Comput Architechture}
{CS 6235 Real Time Systems}
{ECE 6258 Digital Image Processing}

Important links that might help the student.(Buzzport, Course Catalog, Course Critique, Georgia Tech Calender)



Important Links!
Your BuzzPort!!
Find your Course!! - Georgia Tech Course Catalog
Average Course Grade!! - Course Critique
Georgia Tech Calender
Logout
All courses
Recommended

Enter your Term:

Enter your TIG: Computer Systems and Software

Enter your courses taken:

Personal information
Name is: Jennifer
Degree Level: Master's
Field is: ECE
TIG : Computer Systems and Software

Show 25 entries   Search:

tig.ece
ECE 6100 Adv Comput Architechture
ECE 6101 Parallel & Dist Comp Arch
ECE 6102 Dependable Distribut Sys
ECE 6110 CAD â€“ Communication Networks
ECE 6122 Adv Prog Techniques
ECE 7102 RISC Architectures
ECE 7103 Advanced Memory Systems

Deliverables -

- **Dataset Generation using GT Course Catalog – (Chaitanya)** Due to the vast number of courses offered at Georgia Institute of Technology, we limited our focus to the graduate level courses offered in the ECE and CS departments. This involved generating a course dataset consisting of all the courses in that particular department, filtered by specialization, and the semesters that they were offered in.
- **Survey Information from Students – (Sampada)** Carried out a survey on around 100 graduate students from Georgia Tech ECE and CS department to generate a database of the courses opted by students in Fall and Spring semester. This survey was carried out as Georgia Tech student course information was not accessible.
- **Implementation of the Apriori Algorithm on a smaller dataset & scaling it to a larger dataset – (Chaitanya)** The data mining procedure involved studying various algorithms for generation of association rules. While the FP-Growth algorithm is the best in comparison to the Apriori and Eclat algorithm, implementing the FP-Growth in R is slightly more complex than the Apriori algorithm. Up to a couple of tens of thousands of entries, the Apriori algorithm performs well enough and hence was chosen as the data mining tool for rule generation. This involved implementing the algorithm in a smaller subset of the data and tweaking the support and confidence parameters for optimum rule generation. Once the course catalog information and Student registration information was available, the algorithm was scaled appropriately.
- **Building a simple yet functional UI in RShiny (Sampada)** Studied the number of different APIs and packages available, that can be used to develop an interactive web application in R (ex. R Shiny and OpenCPU). Amongst them, R Shiny was selected for being a readily available package in and easy to integrate with R language. Designed a user interface (GUI) with appropriate widgets for the application involved the use of the shiny dashboard package to create the ui.r and server.r file (necessary for running the app in a web browser). Connected the ui.r and server.r file through reactive programming. Integrated the recommendation logic (Apriori algorithm) with the server.r file for the app to be able to recommend courses based on the users' inputs.
- **Building a Login/Registration portal in Shiny (Ishika)** Built and designed a registration page that a new user is prompted to fill out once s/he tries to register into the Course Recommender System. For the first time, it is necessary for the system to input and save the user's personal information into its database so that s/he can be authorized

to log in to the system the next time without having to enter same information again. The database manages information like username and password (for authentication), the student's Name, Field, TIG and current semester. The process includes the updation (during registration, to enter new user data) and sequential search of the database (during login to the match username and password and pull up the rest of the user's details).

- **Database and Association rule updation based on new user information (Ishika)** There are two databases (fall and spring) of former students which are dynamically updated. If it is not the user's first semester and he is currently in the fall semester, the four courses he had taken up in spring are appended to the spring database and likewise. This helps in automatic updation of the database without having to manually enter new data. Additionally, this helps in tracking the current upcoming trends. Association rules are then generated on the updated database each time, therefore, they too change dynamically and reflect current course popularity. This implies and ensures that the recommendation updates in real-time.

Limitations and Future Work -

Firstly, this recommender system uses collaborative filtering for recommendation. Collaborative filtering systems base their recommendation for a given user on the past preferences of other users who share his taste. In the first stage of such a recommendation process, the system suffers from a cold start problem. Successful recommendations for our application depend on the availability of a critical mass of users. This system is not able to recommend courses if no past students database is available.

Secondly, this system does not recommend cross listed courses or prerequisites as they are not included in the courses database.

Another limitation of this recommendation system is that the courses are recommended solely based on their popularity. The algorithm ensures that the courses are recommended only if a majority of the past students have opted for them. Thus, even if a particular course is important for a TIG, it won't be recommended unless it was popular amongst the past students.

We plan on overcoming this limitation by updating the algorithm to recommend courses based on course weights. According to the specialization, all the courses will be labeled with certain weights (eg: from range of 1 to 4). As a consequence, even if the a particular of course with high

weightage (4) is not recommended by the Apriori algorithm (as it is not popular enough), it will automatically get recommended by the updated algorithm.
Right now, this system is able to recommend courses from the departments of ECE and CS in Georgia Tech. In the future, we plan to include the courses from all other departments and universities. Other add-on features such as course rating, feedback system and integrating recommendation with average grades can be implemented. Relevant courses from massive open online courses (MOOCs) can also be recommended using appropriate APIs.

Conclusion -

Universities offer plethora of courses to students studying in different fields which creates a lot of confusion amongst them. Thus majority of the students turn to their peers or advisors for sound recommendations. The proposed course recommender acts as an intermediary system and successfully recommends suitable course to individual students, taking into consideration their field of interest and courses already taken, with the aim of maximizing students' professional foundation. Students mostly opt for the subjects that have been more popular amongst their seniors. Our recommendation algorithm uses this to its advantage by recommending courses that are taken by majority of the past students. This makes it a lot easier for the student to decide on subjects on their own rather than reaching out to their peers and advisors, which might not be always possible. Moreover, this system is self-learning, i.e., it continually updates the student database and updates the rules to accommodate the changing environment of course registration. Thus it successfully addresses and takes into account the dynamics of changing courses before recommending.

References –

[1] Y. L. &. J. Cho, "“An Intelligent Course Recommendation System”," *Smart Computing Review,* vol. 1, no. 1, October 2011.

[2] T. Denley, "Austin Peay State University: Degree Compass," *Game Changers: Education and Information Technologies,* 2012.

[3] M. L. S. a. S. C. Ekdahl, "A student course recommender," 2002.

[4] K. Chu, "Designing a course recommendation system on web based on the students' course selection records," *Proc. of World Conference on Educational Multimedia,* pp. 14-21, 2003.

[5] R. S. R. Agrawal, "Fast algorithms for mining association rules in large databases," *Proc. of the 20th International Conference on Very Large DataBases,* pp. 478-499, 1994.

[6] H. Boydzovska, "Course Enrollment Recommender System," *Educational Conference on Educational Data Mining,* pp. 312 - 317.

[7] S. L. C. S. Magnus Ekdahl, "A Student Course Recommender," 2002.

[8] E. A. Narimel Bendakir, "Using Asscociation Rules for Course Recommendation," *American Association for Artifucial Intelligence,* 2006.

[9] J. K. J. a. R. J. Herlocker, "Explaining collaborative filtering recommandations," in *Proceedings of the ACM 2000 Conference on Computer Supported Cooperative Work*, 2000.

[10] "The 'Shiny' Package," [Online]. Available: https://cran.r-project.org/web/packages/shiny/shiny.pdf.

[11] J. B. Schafer, "The Application of Data Mining to Recommender Systems.," *Encyclopaedia of Data Warehousing and Mining,* pp. 44-48, 2005.

[12] S. P, *Course recommender,* Senior Thesis, Princeton University, 2003.