

Sistema de Recomendação de Disciplinas - “Sistema Matrículas Racket”

Beatriz M. A. Matsui¹, Isaias A. da Silva¹, Renan M. de Oliveira¹

¹Centro de Matemática, Computação e Cognição– Universidade Federal do ABC (UFABC)
Av. dos Estados, 5001 – 09210-580 – Santo André – SP – Brazil

Abstract. *The following project represents the development of a recommender system for Computer Science students from Federal University of ABC (UFABC). For this purpose, the system will consider the user's choices (related to the campus and period), as well as the disciplines already taken by this student, in order to generate the suggestion of the new disciplines. The system relies on the reading of .csv files and the output of another file containing the suggestions of disciplines to the user, all by input and output, in the program language Racket.*

Resumo. *O presente projeto traz a representação de um sistema de recomendação de disciplinas para alunos do curso de Ciência da Computação da Universidade Federal do ABC (UFABC). Para isto, o sistema irá considerar as escolhas do usuário (relacionadas ao câmpus e turno), assim como as disciplinas já cursadas por este aluno, a fim de gerar a sugestão das novas disciplinas. O sistema conta com a leitura de arquivos .csv e a saída de outro arquivo contendo a sugestão de disciplinas para o usuário, tudo mediante entrada e saída, na linguagem de programação Racket.*

1. Introdução

O contexto de desenvolvimento do presente trabalho é o de um sistema de matrículas com muita liberdade e variáveis, como o da oferta de disciplinas, o das disciplinas já cursadas pelo estudante e o curso específico o qual pretende cursar, além da possibilidade de desvios do quadrimestre ideal.

O problema que inspirou a realização deste projeto foi a complexidade para o aluno realizar sua matrícula a cada quadrimestre na UFABC, assim como a possibilidade de cometer erros e os desgastes que sofre durante o processo.

Atualmente, não há ferramentas que auxiliem neste processo - acredita-se que isto ocorra para que o aluno tenha total responsabilidade e consciência da grade que estará montando. Por este motivo, o projeto visa apenas auxiliar e propor uma grade adequada, mantendo a decisão por seguí-la ou não, nas mãos do aluno.

Portanto, pretende-se desenvolver um sistema que receba as disciplinas que o aluno já cursou, e processe as disciplinas disponíveis para o próximo quadrimestre. Com essas informações, também deve considerar a conexão entre as disciplinas e, além disso, o curso específico do aluno (computação, no caso), e lhe forneça uma sugestão otimizada de grade para o quadrimestre seguinte.

1.1. Justificativa

A abordagem adotada no presente projeto é adequada pois fornece ao aluno uma grande personalização para a sua necessidade específica e, também, pois o ajuda a diminuir a chance de escolher uma grade não-ótima para o próximo quadrimestre. Além disso, é uma abordagem original pois ainda não existe um sistema assim para alunos da UFABC e seria difícil utilizar outro sistema de outra universidade, uma vez que a UFABC se diferencia no Brasil por seu projeto pedagógico inovador [UFABC 2006].

2. Definição do Problema

O problema que pretende-se resolver consiste em um *Course Scheduling Program* (CSP) [Gunawan et al. 2007] que envolve o conceito do problema de satisfação de restrições (do inglês *Constraint Satisfaction Problem*) [Gunawan et al. 2007, Haralick and Elliott 1980], presente nos campos de Inteligência Artificial e *Machine Learning*, entre outros. Problemas de *university scheduling* (US) atraem grande atenção por parte de pesquisadores das áreas de IA e pesquisa operacional [Yazdani et al. 2017].

Normalmente, nesses problemas, um conjunto de eventos (que podem ser representados pelos exames ou cursos), é atribuído a um certo número de horários disponíveis para se montar a grade. Em nosso caso, o sistema se baseará nas disciplinas já cursadas pelo aluno e da grade existente de disciplinas no momento (levando-se em conta as disciplinas dos cursos de bacharelado Ciência e Tecnologia (BC&T) e Ciência da Computação (BCC)) para sugerir quais matérias o aluno poderia cursar naquele quadrimestre.

Considerando as peculiaridades da oferta de disciplinas e das grades sugeridas na UFABC, bem como as diferenças de aluno para aluno ao longo de suas trajetórias acadêmicas, nosso sistema se diferencia dos demais que se propõem a resolver problemas semelhantes, apresentando vantagens relacionadas à personalização da grade, levando-se em conta as opções do aluno.

3. Arquitetura e Implementação

O sistema divide-se em dois arquivos principais, escritos na linguagem de programação Racket [Felleisen et al. 2015], são eles: “matriculas-main.rkt” e “menu-rkt”.

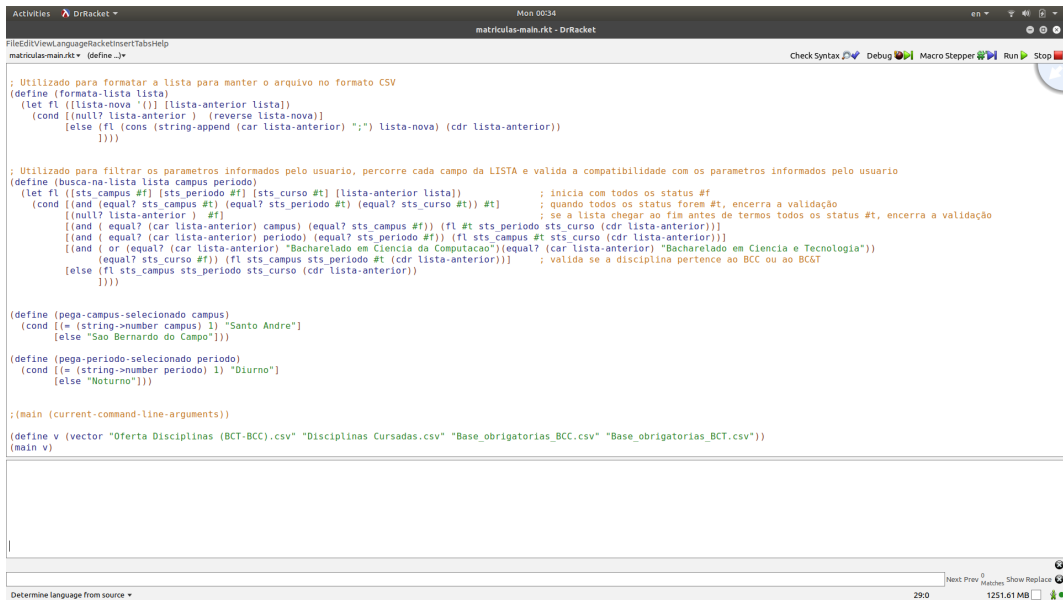
No arquivo “menu.rkt” solicita-se ao usuário que este selecione o câmpus desejado (se Santo André ou São Bernardo) e, em seguida, o período (se Diurno ou Noturno).

No arquivo “matriculas-main.rkt” são criados os procedimentos que realizam as leituras dos arquivos .csv de entrada, os processamentos de comparação entre as bases de disciplinas ofertadas e as cursadas pelo aluno, além de comparar com as disciplinas obrigatórias do BC&T e as obrigatórias somente do BCC.

4. Avaliação

Foram realizados testes com dados reais de um aluno e a oferta de disciplinas da Pró-Reitoria de Graduação da UFABC para o segundo quadrimestre de 2018 utilizando ambos os turnos e ambos os campi. Além disso, foi utilizada a grade curricular do Bacharelado em Ciência da Computação da UFABC.

Será apresentado a seguir um dos testes realizados. O teste para o caso de o aluno realizar suas disciplinas, no quadrimestre em questão, no campus de Santo André no período Diurno:



```
; Utilizado para formatar a lista para manter o arquivo no formato CSV
(define (formata-lista lista)
  (let fl ([lista-nova '()] [lista-anterior lista])
    (cond [(null? lista-anterior) (reverse lista-nova)]
          [else (fl (cons (string-append (car lista-anterior) ",") lista-nova) (cdr lista-anterior))])))

; Utilizado para filtrar os parametros informados pelo usuario, percorre cada campo da LISTA e valida a compatibilidade com os parametros informados pelo usuario
(define (busca-na-lista lista campus periodo)
  (let fl ([sts-campus #f] [sts-periodo #f] [sts-curso #t] [lista-anterior lista])
    (cond [(and (equal? sts-campus #t) (equal? sts-periodo #t) (equal? sts-curso #t)) #t] ; inicia com todos os status #t, encerra a validação
          [(and (equal? sts-campus #f) (equal? sts-periodo #f) (equal? sts-curso #f)) #f] ; quando todos os status forem #f, encerra a validação
          [(and (equal? sts-campus #t) (equal? sts-periodo #f)) (fl #t sts-periodo sts-curso (cdr lista-anterior))] ; se a lista chegar ao fim antes de termos todos os status #t, encerra a validação
          [(and (equal? sts-campus #f) (equal? sts-periodo #t)) (fl #f sts-campus sts-curso (cdr lista-anterior))]
          [(and (equal? sts-campus #t) (equal? sts-periodo #t)) (fl #t sts-campus sts-curso (cdr lista-anterior))]
          [(and (or (equal? (car lista-anterior) "Bacharelado em Ciencia da Computacao") (equal? (car lista-anterior) "Bacharelado em Ciencia e Tecnologia"))
                (equal? sts-curso #f)) (fl sts-campus sts-periodo #t (cdr lista-anterior))] ; valida se a disciplina pertence ao BCC ou ao BCT
          [else (fl sts-campus sts-periodo sts-curso (cdr lista-anterior))])])

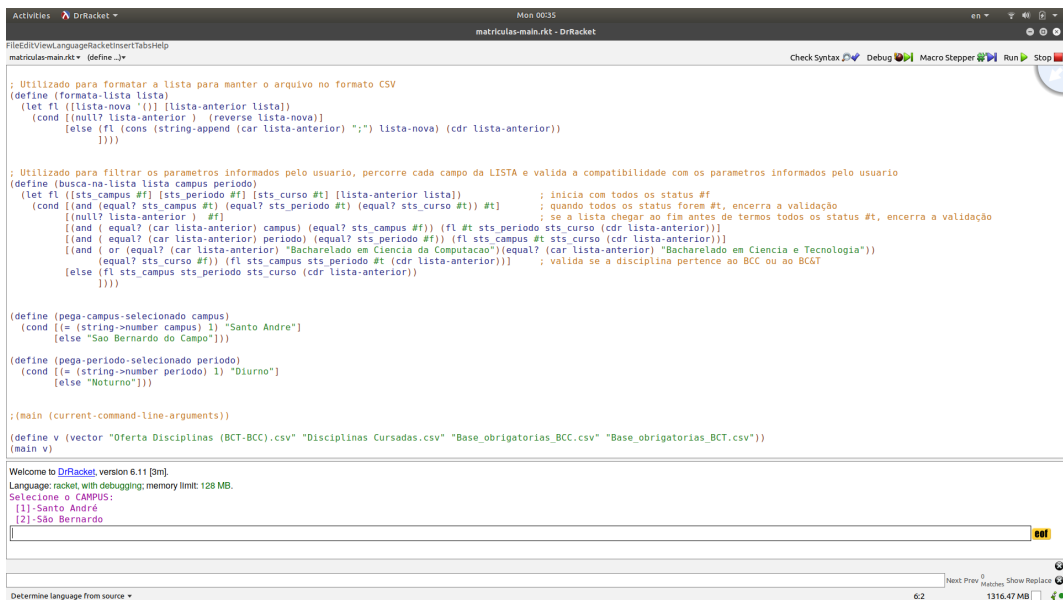
(define (pega-campus-selecionado campus)
  (cond [(= (string-number campus) 1) "Santo Andre"]
        [else "Sao Bernardo do Campo"]))

(define (pega-periodo-selecionado periodo)
  (cond [(= (string-number periodo) 1) "Diurno"]
        [else "Noturno"]))

;(main (current-command-line-arguments))

(define v (vector "Oferta Disciplinas (BCT-BCC).csv" "Disciplinas Cursadas.csv" "Base_obrigatorias_BCC.csv" "Base_obrigatorias_BCT.csv"))
(main v)
```

Figura 1. Sistema antes da execução



```
; Utilizado para formatar a lista para manter o arquivo no formato CSV
(define (formata-lista lista)
  (let fl ([lista-nova '()] [lista-anterior lista])
    (cond [(null? lista-anterior) (reverse lista-nova)]
          [else (fl (cons (string-append (car lista-anterior) ",") lista-nova) (cdr lista-anterior))])))

; Utilizado para filtrar os parametros informados pelo usuario, percorre cada campo da LISTA e valida a compatibilidade com os parametros informados pelo usuario
(define (busca-na-lista lista campus periodo)
  (let fl ([sts-campus #f] [sts-periodo #f] [sts-curso #t] [lista-anterior lista])
    (cond [(and (equal? sts-campus #t) (equal? sts-periodo #t) (equal? sts-curso #t)) #t] ; inicia com todos os status #t, encerra a validação
          [(and (equal? sts-campus #f) (equal? sts-periodo #f) (equal? sts-curso #f)) #f] ; quando todos os status forem #f, encerra a validação
          [(and (equal? sts-campus #t) (equal? sts-periodo #f)) (fl #t sts-periodo sts-curso (cdr lista-anterior))] ; se a lista chegar ao fim antes de termos todos os status #t, encerra a validação
          [(and (equal? sts-campus #f) (equal? sts-periodo #t)) (fl #f sts-campus sts-curso (cdr lista-anterior))]
          [(and (equal? sts-campus #t) (equal? sts-periodo #t)) (fl #t sts-campus sts-curso (cdr lista-anterior))]
          [(and (or (equal? (car lista-anterior) "Bacharelado em Ciencia da Computacao") (equal? (car lista-anterior) "Bacharelado em Ciencia e Tecnologia"))
                (equal? sts-curso #f)) (fl sts-campus sts-periodo #t (cdr lista-anterior))] ; valida se a disciplina pertence ao BCC ou ao BCT
          [else (fl sts-campus sts-periodo sts-curso (cdr lista-anterior))])])

(define (pega-campus-selecionado campus)
  (cond [(= (string-number campus) 1) "Santo Andre"]
        [else "Sao Bernardo do Campo"]))

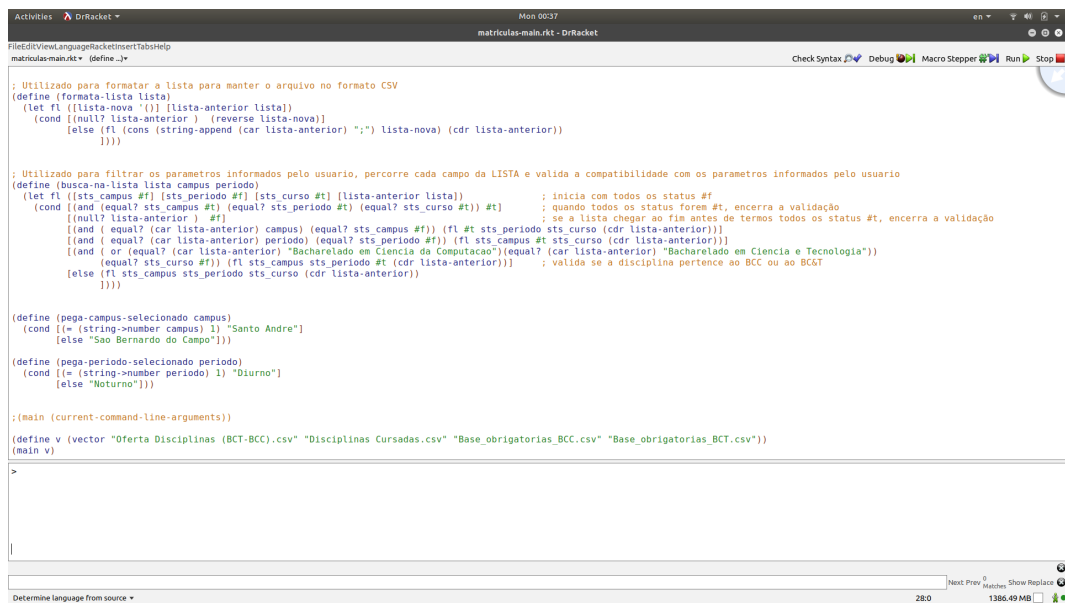
(define (pega-periodo-selecionado periodo)
  (cond [(= (string-number periodo) 1) "Diurno"]
        [else "Noturno"]))

;(main (current-command-line-arguments))

(define v (vector "Oferta Disciplinas (BCT-BCC).csv" "Disciplinas Cursadas.csv" "Base_obrigatorias_BCC.csv" "Base_obrigatorias_BCT.csv"))
(main v)
```

Welcome to DrRacket, version 6.11 [6m].
Language: racket, with debugging; memory limit: 128 MB.
Selecione o CAMPUS:
[1]-Santo André
[2]-São Bernardo

Figura 2. Escolhendo o parâmetro Campus



```
; Utilizado para formatar a lista para manter o arquivo no formato CSV
(define (formato-lista lista)
  (let fl ([lista-nova '()] [lista-anterior lista])
    (cond [(null? lista-anterior) (reverse lista-nova)]
          [else (fl (cons (string-append (car lista-anterior) ";") lista-nova) (cdr lista-anterior))]))))

; Utilizado para filtrar os parametros informados pelo usuario, percorre cada campo da LISTA e valida a compatibilidade com os parametros informados pelo usuario
(define (busca-na-lista lista campus periodo)
  (let fl ([sts-campus #f] [sts-periodo #f] [lista-anterior lista])
    (cond [(and (equal? sts-campus #t) (equal? sts-periodo #t) (equal? sts-curso #t)) #t] ; inicia com todos os status #f
          [(null? lista-anterior) #f] ; quando todos os status forem #t, encerra a validação
          [(and (equal? (car lista-anterior) campus) (equal? sts-campus #f)) (fl #t sts-periodo sts-curso (cdr lista-anterior))]] ; se a lista chegar ao fim antes de termos todos os status #t, encerra a validação
          [(and (equal? (car lista-anterior) periodo) (equal? sts-periodo #f)) (fl sts-campus sts-curso (cdr lista-anterior))]]
          [(and (or (equal? (car lista-anterior) "Bacharelado em Ciencia da Computacao") (equal? (car lista-anterior) "Bacharelado em Ciencia e Tecnologia"))
                  (equal? sts-curso #f)) (fl sts-campus sts-periodo #t (cdr lista-anterior))]] ; valida se a disciplina pertence ao BCC ou ao BCT
          [else (fl sts-campus sts-periodo sts-curso (cdr lista-anterior))]]))

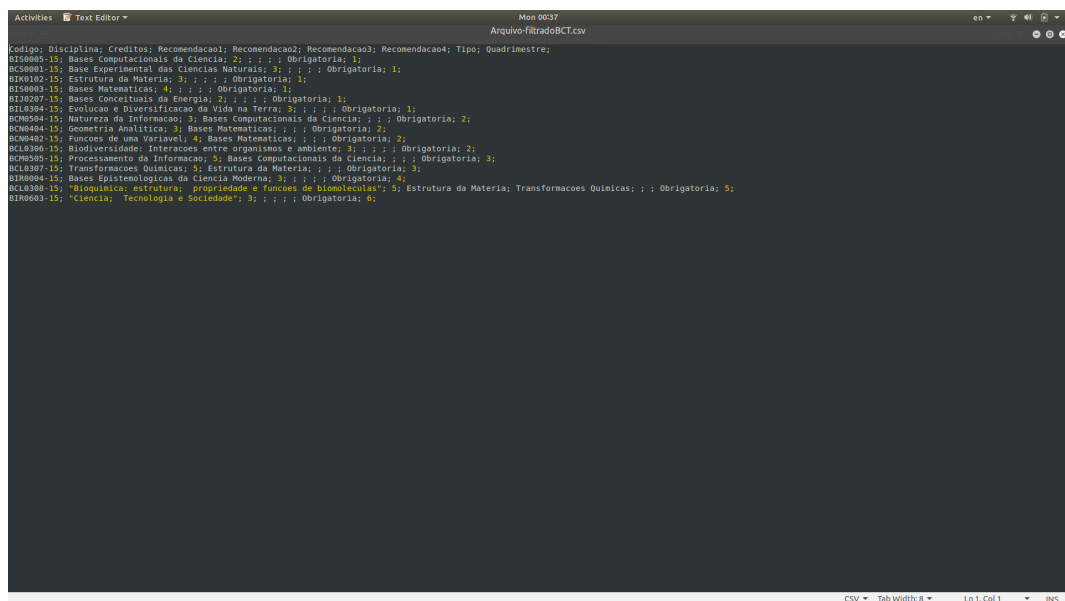
(define (pega-campus-selecionado campus)
  (cond [(= (string->number campus) 1) "Santo Andre"]
        [else "Sao Bernardo do Campo"]))

(define (pega-periodo-selecionado periodo)
  (cond [(= (string->number periodo) 1) "Diurno"]
        [else "Noturno"]))

;(main (current-command-line-arguments))

(define v (vector "Oferta Disciplinas (BCT-BCC).csv" "Disciplinas Cursadas.csv" "Base_obrigatorias_BCC.csv" "Base_obrigatorias_BCT.csv"))
(main v)
```

Figura 3. Escolhendo o parâmetro Turno



```
codigo; Disciplina; Creditos; Recomendacao1; Recomendacao2; Recomendacao3; Recomendacao4; Tipo; Quadrimestre;
BIS0005-15; Bases Computacionais da Ciencia; 2; ; ; ; Obrigatoria; 1;
BCS0001-15; Base Experimental das Ciencias Naturais; 3; ; ; ; Obrigatoria; 1;
BIK0102-15; Estrutura da Matéria; 3; ; ; ; Obrigatoria; 1;
BIS0003-15; Bases Matematicas; 4; ; ; ; Obrigatoria; 1;
BIJ0207-15; Bases Conceituais da Energia; 2; ; ; ; Obrigatoria; 1;
BIL0304-15; Evolucao e Diversificacao da Vida na Terra; 3; ; ; ; Obrigatoria; 1;
BCW0504-15; Natureza da Informacao; 3; Bases Computacionais da Ciencia; ; ; ; Obrigatoria; 2;
BCN0404-15; Geometria Analitica; 3; Bases Matematicas; ; ; ; Obrigatoria; 2;
BCN0402-15; Funcoes de uma Variavel; 4; Bases Matematicas; ; ; ; Obrigatoria; 2;
BCL0306-15; Biodiversidade: Interacoes entre organismos e ambiente; 3; ; ; ; Obrigatoria; 2;
BCW0505-15; Processamento da Informacao; 3; Bases Computacionais da Ciencia; ; ; ; Obrigatoria; 3;
BCL0307-15; Transformacoes Quimicas; 3; Estrutura da Matéria; 2; ; ; Obrigatoria; 3;
BIR0004-15; Bases Epistemologicas da Ciencia Moderna; 3; ; ; ; Obrigatoria; 4;
BCL0308-15; "Bioquímica: estrutura, propriedades e funcoes de biomoléculas"; 5; Estrutura da Matéria; Transformacoes Quimicas; ; ; ; Obrigatoria; 5;
BIS0003-15; "Ciencia, Tecnologia e Sociedade"; 3; ; ; ; Obrigatoria; 6;
```

Figura 4. Arquivo com as disciplinas obrigatórias do BCT para o quadrimestre

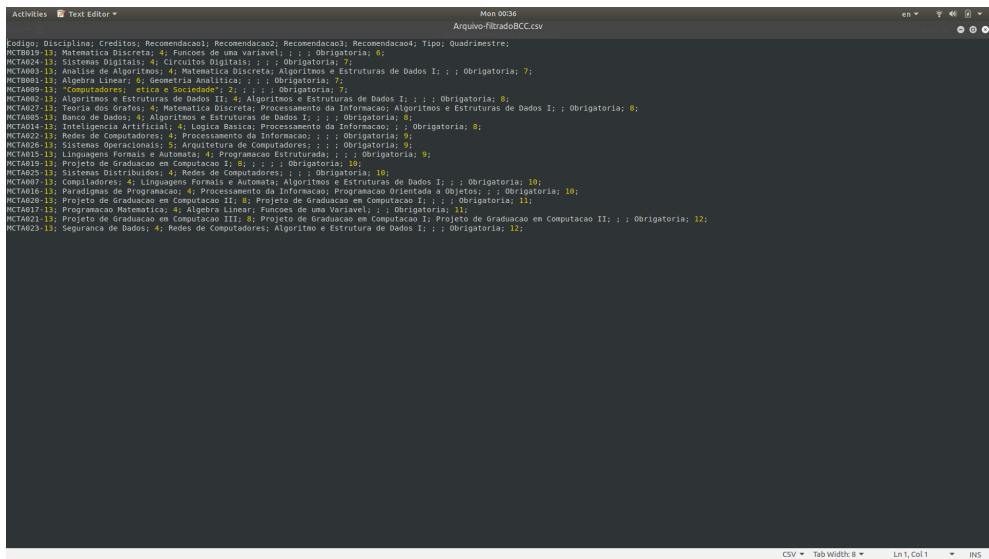


Figura 5. Arquivo com as disciplinas obrigatórias do BCC para o quadrimestre



Figura 6. Arquivo com as disciplinas do Campus e Turno escolhidos

5. Discussão

O projeto aqui apresentado buscou aplicar os conhecimentos aprendidos durante o segundo quadrimestre de 2018 na disciplina Paradigmas de Programação, sobretudo no que se refere ao paradigma funcional e à linguagem de programação Racket.

Com uma temática voltada à universidade, à organização e ao planejamento de matrícula, foi possível tornar ainda mais próximo da realidade do grupo o desenvolvimento deste projeto, tornando o tema agradável e estimulante de se trabalhar.

O grupo encontrou dificuldades principalmente em relação ao paradigma funcional (pois foi a primeira vez que tiveram contato com um paradigma diferente do imperativo).

Foi preciso realizar mudanças no projeto ao longo do seu desenvolvimento (por exemplo, deixar de utilizar banco de dados e passar a tratar os dados somente com arquivos de entrada e saída) - tópico que os membros do grupo aprenderam na disciplina bem no momento em que estavam em dúvida se utilizariam banco de dados ou não.

Entretanto, esta foi uma mudança boa, pois o grupo pôde aplicar ainda mais o que foi aprendido em sala de aula e laboratório na realização do projeto.

Além disso, pode-se dizer que muito foi aprendido, não só em termos de computação, como também em gerenciamento de projetos, trabalho em equipe e resiliência.

6. Contribuição de Cada Integrante

Nesta seção, detalhamos as participações de cada integrante da equipe no desenvolvimento do projeto:

Beatriz - responsável pela criação e organização do repositório do projeto no Github (vide link na Seção 7), bem como pela codificação do filtro e geração de arquivo de disciplinas para o usuário com base nas disciplinas obrigatórias do BCC. Divisão entre os arquivos “menu.rkt” e “matriculas-main.rkt”. Responsável pela manutenção dos versionamentos e testes, e pela escrita da documentação.

Isaias - responsável pela codificação da estrutura base: código *main* contendo recebimento dos arquivos .csv de entrada e saída para o usuário, menu, geração dos arquivos de entrada (convertidos para csv a partir de arquivos pdf).

Renan - responsável por codificar, testar, implementar melhorias e corrigir problemas no código, além de implementar o filtro das disciplinas obrigatórias do BCT e de gerar o arquivo da grade do BCC. Responsável também pela elaboração e gravação do vídeo contendo a explicação do funcionamento do sistema, além da documentação.

7. Repositório no Github

https://github.com/beatrizmayumi/sistema_matriculas

Referências

- [Felleisen et al. 2015] Felleisen, M., Findler, R. B., Flatt, M., Krishnamurthi, S., Barzilay, E., McCarthy, J., and Tobin-Hochstadt, S. (2015). The racket manifesto. In *LIPICs-Leibniz International Proceedings in Informatics*, volume 32. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.
- [Gunawan et al. 2007] Gunawan, A., Ng, K. M., and Poh, K. L. (2007). Solving the teacher assignment-course scheduling problem by a hybrid algorithm. *International Journal of Computer, Information, and Systems Science, and Engineering*, 1(2):136.
- [Haralick and Elliott 1980] Haralick, R. M. and Elliott, G. L. (1980). Increasing tree search efficiency for constraint satisfaction problems. *Artificial intelligence*, 14(3):263–313.
- [UFABC 2006] UFABC (2006). Projeto pedagógico da universidade federal do abc.
- [Yazdani et al. 2017] Yazdani, M., Naderi, B., and Zeinali, E. (2017). Algorithms for university course scheduling problems. *Tehnicki Vjesnik-Technical Gazette*, 24:241–247.