INSTITUTO FEDERAL Espírito Santo Campus Serra	**Curso de Engenharia de Controle e Automação**			
	Componente Curricular: Programação Orientada a Objetos			
	Professor: Hilário Tomaz Alves de Oliveira			
	Semestre: 2023.2	**Período:** 3°	**Turma:** Noite	**Data Entrega:** 04/12/2023

Projeto de Implementação - Sistema de Controle Acadêmico

1. Descrição

O objetivo deste projeto é implementar um sistema de controle acadêmico usando a linguagem de Programação Python. O sistema vislumbrado deve permitir o controle de disciplinas e alunos para uma instituição de ensino. Inicialmente, é necessário criar um banco de dados no **SQLite**, com o nome **`controle_academico.db`,** contendo as tabelas **`Disciplina`**, **`Aluno`** e **`Matricula`**. Os campos dessas tabelas serão descritos ao longo deste documento. O banco de dados deve ser criado somente na primeira execução o sistema. O sistema desenvolvido deve possuir o seguinte menu principal:

```
---------------- Menu Principal ----------------

  1 - Controle de Disciplinas
  2 - Controle de Alunos(as)
  3 - Controle de Matrículas
  4 - Sair


Digite sua Opção:
```

- Caso a **OPÇÃO 1** seja selecionada, o seguinte meu deve ser exibido:

```
---------------- Controle de Disciplinas ----------------

     1 - Cadastrar Nova Disciplina
     2 - Atualizar Disciplina
     3 - Remover Disciplina
     4 - Listar Disciplinas
     5 - Voltar Menu Principal

Digite sua Opção:
```

  - Se a **OPÇÃO 1** for escolhida, o sistema deve solicitar do usuário o **`código da disciplina`**, o **`nome da disciplina`**, a **`carga horária`** e o **`nome do professor responsável`**.

    - **OBS 1:** O cadastro da disciplina deve ser feito no banco de dados na Tabela **Disciplina**.

    - **OBS 2:** Seu programa **não deve permitir** que duas disciplinas com o mesmo **`código`** sejam cadastradas.

  - Se a **OPÇÃO 2** for escolhida, o programa deve solicitar o **`código da disciplina`** que se deseja atualizar. Caso seja digitado um **`código`** de uma disciplina que não existe no banco de dados, deve-se exibir uma mensagem avisando. Caso a disciplina exista no banco de dados, deve-se exibir os dados atuais da disciplina (**`código`**, **`nome`**, **`carga horária`** e **`professor responsável`**) e solicitar que sejam digitados os novos dados. O atributo **`código`** **não pode ser atualizado**. Caso o usuário não deseje atualizar algum campo basta deixá-lo em branco (Apertar **Enter**). Logo, deve-se manter o valor atual do atributo.

  - Se a **OPÇÃO 3** for escolhida, o programa deve solicitar o **`código`** da disciplina a ser removida e deve deletar a disciplina do banco de dados.

Caso seja digitado um código de uma disciplina que não existe no banco de dados, deve-se exibir uma mensagem avisando.

- Se a **OPÇÃO 4** for escolhida, o sistema deve listar todas as disciplinas cadastradas no sistema. Deve-se exibir o total de disciplinas cadastradas e para cada disciplina exibir o **`código`**, **`nome`**, **`carga horária`** e o **`nome do professor responsável`**. Além disso, o sistema deve perguntar se o usuário deseja exportar os dados das disciplinas em formato *Comma-Separated Values* (CSV). O arquivo deve ser estruturado com cada disciplina em uma linha e os atributos como colunas separadas por ponto-e-vírgula.

- Se a **OPÇÃO 5** for escolhida, o sistema de voltar ao **Menu Principal**.

- Caso a **OPÇÃO 2** seja selecionada, o seguinte meu deve ser exibido:

```
---------------- Controle de Alunos(as) ----------------

     1 – Cadastrar Aluno(a)

     2 – Atualizar Aluno(a)

     3 – Remover Aluno(a)

     4 – Listar Alunos(as)

     5 – Voltar Menu Principal

Digite sua Opção:
```

  - Se a **OPÇÃO 1** for escolhida, o sistema deve solicitar do usuário o **`nome do aluno`**, o **`CPF`**, **`idade, e-mail`** e o **`endereço`**.

    - **OBS 1:** O cadastro do(a) aluno(a) deve ser feito no banco de dados na Tabela **Aluno**.

- **OBS 2:** Seu programa **não deve permitir** que dois (duas) alunos(as) com o mesmo CPF cadastrados (as).

- **OBS 3:** O campo endereço deve ser uma string única formada pela rua (logradouro), bairro, CEP, cidade e estado onde o(a) aluno(a) reside. Para isso, solicite do usuário o CEP do endereço e seu programa deve consultar o seguinte serviço Web para obter o endereço completo a partir do CEP:

    - Serviço: https://viacep.com.br/ws/**CEP**/json/

        - Trocar **CEP** pelo valor do CEP digitado pelo usuário (sem pontuação).

        - Por exemplo (cole no seu navegador para ver o resultado): https://viacep.com.br/ws/**29166630**/json/

        - Utilize a biblioteca **requests**[1] para fazer uma requisição a URL do serviço. O resultado é um arquivo no formato JSON que pode ser facilmente convertido para um dicionário em Python.

            - Pesquise como fazer isso.

    - Caso não seja possível encontrar as informações pelo CEP, o sistema deve solicitar para o usuário o endereço completo via teclado.

- Se a **OPÇÃO 2** for escolhida, o programa deve solicitar o **`CPF do aluno(a)`** que se deseja atualizar. Caso seja digitado um **`CPF`** de um(a) aluno(a) que não existe no banco de dados, deve-se exibir uma mensagem avisando. Caso o(a) aluno(a) exista no banco de dados, deve-se exibir os

[1] https://requests.readthedocs.io/en/latest/

dados atuais do aluno(a) (**nome**, o **CPF**, **idade**, **e-mail** e **endereço**) e solicitar que sejam digitados os novos dados. O CPF do(a) aluno(a) **não deve ser solicitado para atualização.** Caso o usuário não deseje atualizar algum campo basta deixá-lo em branco (Apertar **Enter**). Logo, deve-se manter o valor atual do atributo.

- Se a **OPÇÃO 3** for escolhida, o programa deve solicitar o **CPF** do aluno(a) a ser removido(a) e deve deletar do banco de dados. Caso seja digitado um **CPF** de um(a) aluno(a) que não existe no banco de dados, deve-se exibir uma mensagem avisando.

- Se a **OPÇÃO 4** for escolhida, o sistema deve listar todos(as) os(as) alunos(as) cadastrados(as) no sistema. O sistema deve exibir o total de alunos(as) cadastrados(as) e para cada aluno(a) deve-se exibir o **nome**, o **CPF**, **idade**, **e-mail** e **endereço**. Além disso, o sistema deve perguntar se o usuário deseja exportar os dados dos(as) alunos(as) em formato *Comma-Separated Values* (CSV). O arquivo deve ser estruturado com cada aluno(a) em uma linha e os atributos como colunas separadas por ponto-e-vírgula.

- Se a **OPÇÃO 5** for escolhida, o sistema de voltar ao **Menu Principal**.

- Caso a **OPÇÃO 3** seja selecionada, o seguinte meu deve ser exibido:

```
---------------- Controle de Matrículas ----------------

     1 - Matricular Aluno(a)

     2 - Cancelar Matrícula

     3 - Listar Matrículas

     4 - Voltar Menu Principal

Digite sua Opção:
```

- Se a **OPÇÃO 1** for escolhida, o sistema deve solicitar do usuário o **`código da disciplina`** e o **`CPF do(a) aluno`** que deseja matricular. **Seu sistema só deve permitir a matrícula de um(a) aluno(a) em uma disciplina se ambas estejam previamente cadastradas no banco de dados**. Caso contrário, deve-se exibir uma mensagem de erro. Caso o(a) aluno(a) e a disciplina estejam previamente inseridas no banco de dados, deve-se automaticamente registra a **`data e o horário da matrícula`**.

    - **OBS 1:** O cadastro da matrícula deve ser feito no banco de dados na Tabela **Matricula** com os atributos: **`CPF do aluno(a), código da disciplina e data e horário da matrícula`**.

    - **OBS 2:** Seu programa **não deve permitir que o mesmo aluno(a) seja matriculado duas vezes na mesma disciplina**.

- Se a **OPÇÃO 2** for escolhida, o programa deve solicitar o **`CPF do aluno(a)`** e o **`código da disciplina`** da matrícula a ser cancelada. Caso não seja encontrada no banco de dados nenhuma matrícula entre **o(a) aluno(a) do CPF** digitado na **disciplina de código informado**, deve-se exibir uma mensagem avisando, caso contrário, deve-se remover a matrícula do banco de dados.

- Se a **OPÇÃO 3** for escolhida, o sistema deve listar todas as matrículas cadastradas no sistema. O sistema deve exibir o total de matrículas cadastradas e para cada matrícula deve-se exibir o **`CPF do aluno(a), código da disciplina e data e horário da matrícula`**. Além disso, o sistema deve perguntar se o usuário deseja exportar os dados das matrículas em formato *Comma-Separated Values* (CSV). O arquivo deve ser estruturado com cada matrícula em uma linha e os atributos como colunas separadas por ponto-e-vírgula.

- Se a **OPÇÃO 4** for escolhida, o sistema de voltar ao **Menu Principal**.

- Por fim, caso a **OPÇÃO 4** seja selecionada, o sistema deve ser encerrado.

2. Artefatos Entrega

O aluno (ou grupo) deve entregar o *software* especificado neste documento desenvolvido na Linguagem de Programação Python, em uma pasta compactada (.zip ou .rar) que contenha os arquivos fontes. Para isso, crie um projeto (pasta) somente para este sistema. Dentro do projeto crie uma pasta chamada de "**src**" e coloque todos os arquivos de código-fonte dentro dessa pasta. Caso desejem, vocês podem criar subpastas dentro da pasta "**src**". Apenas um aluno do grupo deve enviar a pasta "**src**" compactada usando a tarefa criada no AVA.

3. Requisitos Não Funcionais

Os seguintes requisitos não funcionais deverão pautar todo o projeto e desenvolvimento:

a) É OBRIGATÓRIO o uso dos conceitos de Programação Orientada a Objetos (POO) vistos em sala. Não é necessário usar todos os conceitos, mas pelo menos os conceitos de classes, atributos e métodos devem ser utilizados.

b) A persistência dos dados deve ser feita usando o banco de dados SQLITE.

c) O código-fonte deve ser bem organizado.

d) Os identificadores de variáveis, funções, classes, atributos e métodos devem ser significativos (evite nomes como x1, a2, etc. prefira nome, preco, artigo, entre outros).

e) Coloque um comentário contendo a identificação dos programadores (membros da equipe) no início de cada arquivo de código-fonte.

f) A interface de utilização do sistema deve ser simples, porém, amigável.

g) Projetos copiados serão automaticamente ZERADOS.

4. Prazos e outras informações

- Equipe de, no máximo, 03 (três) alunos.
- O projeto terá uma pontuação máxima de 35,00 pontos.
- A data final para entrega será 04/12/2023 até as 18:00 (via Ambiente Virtual). Apenas um membro da equipe precisa enviar o projeto.
    - OBS: Lembre-se de colocar o nome dos membros da equipe.
- Todos os membros da equipe devem estar presentes na apresentação do projeto, cujas as datas serão nos dias 04/12/2023 e 11/12/2023.
    - Posteriormente, será definido a ordem de apresentação dos trabalhos.