OBJETIVO GENERAL:

Diseñar e implementar una aplicación de Gestión Académica para una Universidad que permita repasar todos los conceptos aprendidos en cursos anteriores.

OBJETIVOS ESPECIFICOS:

- Repasar los fundamentos de Kotlin (Tipos de datos, eventos y operadores)
- Manejar correctamente la herencia, paquetes, interfaces y las bibiotecas.
- Implementar haciendo uso correcto de Capas y Patrones de Diseño.
- Implementar haciendo uso de Manejo de colecciones, paquetes y bibliotecas.

DESCRIPCION:

Una Universidad requiere un sistema de gestión académica que le permita procesar la información de las carreras que ofrece, los cursos que las forman, los profesores que los imparten y los alumnos inscritos. También requiere "programar" cada ciclo lectivo, para lo cual debe abrir el ciclo como tal, registrar la oferta académica, es decir los cursos y grupos que se van a abrir y los profesores que los van a impartir. De igual modo requiere ejecutar el proceso de matrícula, en el cual se registra los cursos, y grupos específicos, que matricula cada estudiante. Por último requiere hacer el proceso final del ciclo, en el cual el profesor registra las notas que obtuvieron los estudiantes en sus grupos.

Información Requerida

A continuación se detalla la información requerida para cada entidad

1. Carrera: código, nombre, título y lista de cursos que la forman, indicando para cada uno el año y ciclo en que debería llevarse.
2. Curso: código, nombre, créditos y horas semanales **, año, ciclo.**
3. Profesor: cédula, nombre, teléfono e email.
4. Alumno: cédula, nombre, teléfono, email, fecha de nacimiento y carrera en que está inscrito. Un alumno solo puede matricular cursos de la carrera en que está inscrito.
5. Ciclo: año, número (1ero o 2ndo), fecha de inicio y fecha de finalización

6. Grupo: ciclo, curso, número de grupo (dentro del ciclo y curso), horario, profesor que lo imparte y lista de estudiantes matriculados en dicho grupo, que luego se completará con la nota que asigne el profesor.
7. Usuarios: Los administradores y matriculadores requieren, al igual que los profesores y alumnos, una cédula y una clave.

Funcionalidades esperadas

1. Mantenimiento de cursos (búsqueda por nombre, código y por carrera).
2. Mantenimiento de carreras (búsqueda por nombre y código), al editar una carrera debe poderse dar mantenimiento a la lista de cursos que la forman, agregando o quitando cursos o cambiándoles el orden.
3. Mantenimiento de Profesores (búsqueda por nombre y cédula).
4. Mantenimiento de Alumnos (búsqueda por nombre, cédula y carrera). Desde el mantenimiento de alumnos debe poderse consultar le historial de dicho alumno.
5. Mantenimiento de ciclos (búsqueda por año). Debe además poderse seleccionar un ciclo como el ciclo activo, ese será el ciclo "default" al preparar la oferta académica, al matricular y al registrar notas.
6. Oferta académica. Debe seleccionarse la carrera y el ciclo y el sistema le mostrará la lista de cursos. Luego podrá seleccionarse un curso y el sistema le mostrará la lista de grupos programados, desde donde podrá agregar o modificar grupos.
7. Matricula: Desde el mantenimiento de estudiantes podrá buscarse el estudiante deseado y seleccionar la opción de matrícula. El sistema le mostrará la lista de cursos matriculados por el estudiante en el ciclo activo y le permitirá agregar o eliminar cursos (grupos). Si el usuario desea puede cambiar a otro ciclo distinto al actual.
8. Registro de notas. El profesor ingresa y el sistema le muestra la lista de cursos (grupos) que tiene a cargo en el ciclo actual, puede seleccionar uno y el sistema le mostrará la lista de estudiantes matriculados y le permitirá registra y/o modificar la nota de cada uno.
9. Consulta de historial. Un alumno podrá ingresar y ver su historial académico
10. Seguridad: mantenimiento de administradores y matriculadores.

Perfiles de usuario

En el sistema habrá los siguientes perfiles de usuario, con los respectivos derechos:

1. Administrador: tiene acceso a todas las funcionalidades, excepto a la 8.
2. Matriculador: tiene acceso a la funcionalidad 7.
3. Profesor: tiene acceso a la funcionalidad 8.
4. Alumno: tiene acceso a la funcionalidad 9.

INSTRUCCIONES GENERALES:

- No se aceptarán trabajos después de la fecha indicada, excepto por razones debidamente justificadas (comprobante).
- Se habilitará en el aula virtual las opciones para que los estudiantes puedan entregar el laboratorio respectivo. Si por alguna razón el profesor pasa la fecha de la entrega de alguna evaluación, lo notificará por medio de la plataforma.
- El laboratorio debe mostrar la eficiencia de un trabajo hecho por un futuro profesional. Por lo tanto se esperan implementaciones bien hechas y que demuestren dominio de los temas solicitados.
- Los trabajos deben ser diferentes, en caso de plagio del trabajo o descarga de internet sin la debida justificación, los estudiantes serán sancionados según el reglamento establecido para tales efectos por la universidad.
- En caso de que el (la) o los (las) estudiantes no puedan demostrar que realizaron el laboratorio, pierden el laboratorio y el mismo será anulado.
- El laboratorio debe estar funcional, de lo contrario no será evaluado. Por lo tanto en su defecto se procederá a poner una nota de CERO.

INSTRUCCIONES ESPECIFICAS:

- El laboratorio se deberá realizar en grupos de estudiantes, según la distribución del profesor.
- A la hora de la entrega del laboratorio, el (la) o los (las) estudiante(s) deberán subir un solo trabajo, el cual debe tener los números de las cédulas respectivos. Por ejemplo: Ced-1…Ced-N.zip.
- No se aceptarán laboratorios enviados fuera de la plataforma y fuera de la fecha de entrega.
- Deben realizar un solo back-end y dos front-end:
    - Web
    - Móvil (será visto en el transcurso del curso, utilizando en el front-end kotlin).
- Cualquier otro elemento que el profesor considere pertinente.

EVALUACION:

Rubro por evaluar	Total porcentaje	Porcentaje obtenido
BackEnd • Base de Datos (funciones y procedimientos/PL-SQL). (20%). • Arquitectura por Capas (Modelo a 5 capas). (15%). • Patrones de diseño (Observer, Sigleton y Iterator). (15%).	**40%**	
FrontEnd • Navegabilidad del sistema (20%). • Manejo adecuado de la Interfaz gráfica (15%) • Uso correcto de cookies, sessions, u otros recursos para la administración de las interfaces gráficas. (15%).	**60%**	
PORCENTAJE TOTAL OBTENIDO	**100%**	