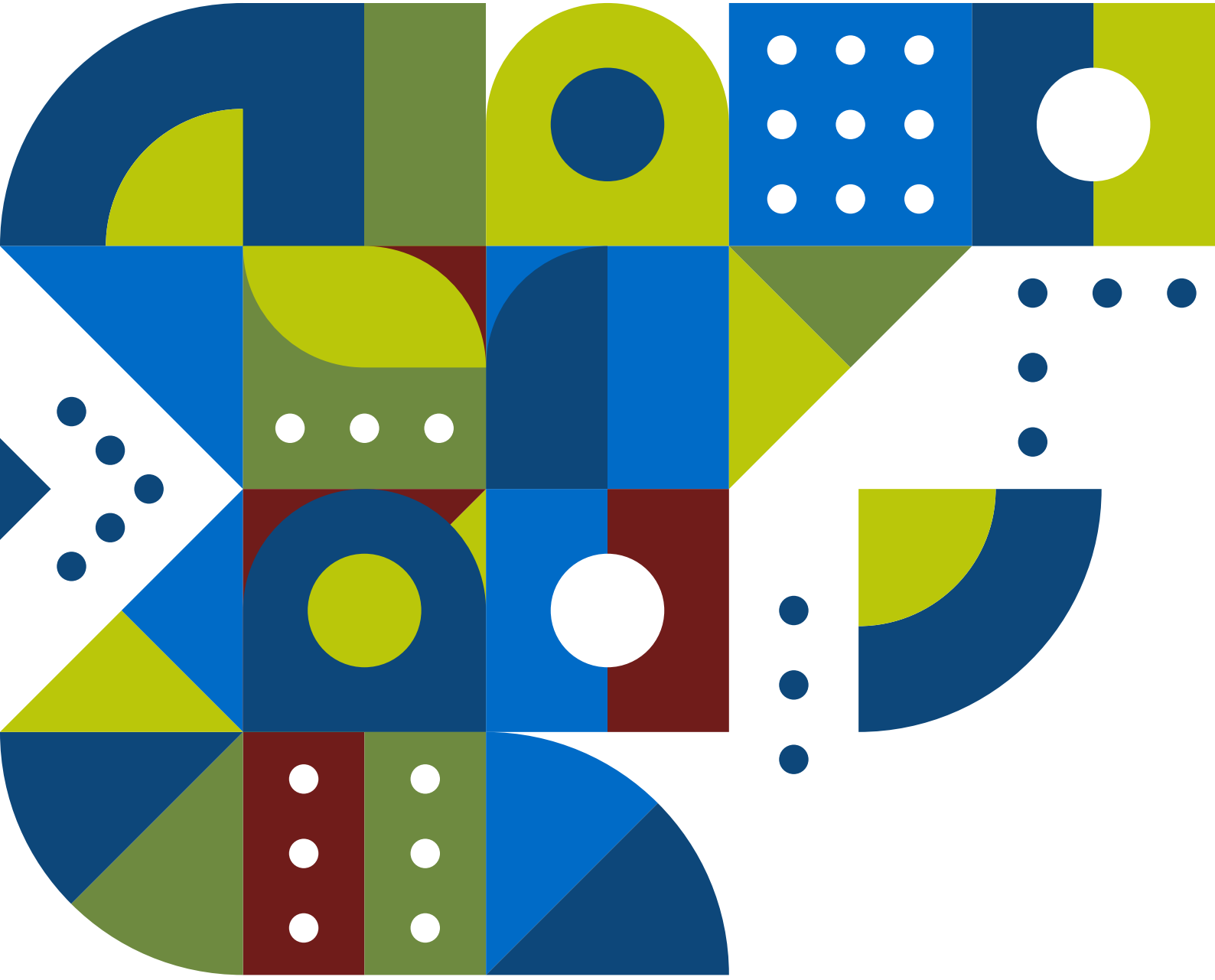
SERIE
Educación Digital, Programación y Robótica

Informática

Crear soluciones: ¿cómo construir un sistema a medida?



Jefe de Gobierno
Horacio Rodríguez Larreta

Ministra de Educación
María Soledad Acuña

Jefe de Gabinete
Manuel Vidal

Subsecretaria de Coordinación Pedagógica y Equidad Educativa
María Lucía Feced Abal

Subsecretario de Carrera Docente
Oscar Mauricio Ghillione

Subsecretario de Tecnología Educativa y Sustentabilidad
Santiago Andrés

Subsecretario de Gestión Económico Financiera y Administración de Recursos
Sebastián Tomaghelli

Subsecretaria de la Agencia de Aprendizaje a lo Largo de la Vida
Eugenia Cortona

Directora Ejecutiva de la Unidad de Evaluación Integral de la Calidad y Equidad Educativa
Carolina Ruggero

Directora General de Educación de Gestión Privada
María Constanza Ortiz

Director General de Planeamiento Educativo
Javier Simón

Directora General de Educación Digital
Julia Campos

Gerente Operativo de Currículum
Eugenio Visiconde

Gerenta Operativa de Tecnología e Innovación Educativa
Sandra Coronel

G.C.A.B.A. | Ministerio de Educación | Dirección General de Planeamiento Educativo | Gerencia Operativa de Currículum.

Dirección General de Planeamiento Educativo (DGPLEDU)
Gerencia Operativa de Currículum (GOC)
Eugenio Visiconde

Coordinadora general: Mariana Rodríguez.

Equipo de especialistas en didáctica del Nivel Secundario: Bettina Bregman (coordinación), Cecilia Bernardi, Ana Campelo, Mariana Gild, Marta Libedinsky, Adriana Vanin.

Especialistas: Marta Libedinsky (coordinación de la serie), Adriana Vanin, Sebastián Frydman.

Subsecretaría de Tecnología Educativa y Sustentabilidad (SSTES)
Dirección General de Educación Digital (DGED)
Gerencia Operativa Tecnología e Innovación Educativa (INTEC)
Sandra Coronel

Especialistas de Educación Digital: Pamela Catarin.

Agradecimientos: a Roberto Tassi, Julia Campos, Ignacio Spina, Josefina Gutierrez por su gestión y acompañamiento durante el proceso de diseño y escritura del documento curricular.

Equipo Editorial de Materiales y Contenidos Digitales (DGPLEDU)
Coordinación general: Silvia Saucedo.
Coordinación editorial: Marcos Alfonzo.
Asistencia editorial: Leticia Lobato.
Edición y corrección: Marta Lacour.
Diseño gráfico y diagramación: Marcela Jiménez.
Producción audiovisual: Joaquín Simón (coordinación y edición de video), Ariel Alvira, Nicolás Bustamante (edición de video), Alejandro Gómez Ferrero (edición de audio y musicalización).

Imágenes: Freepik, Pexels.

ISBN: en trámite.

Presentación

La serie Educación Digital, Programación y Robótica contiene diversas propuestas de enseñanza para el desarrollo de los contenidos, conceptos, capacidades, prácticas, valores y actitudes, definidos en el Diseño Curricular de la NES y en el Anexo Curricular de Educación Digital, Programación y Robótica, Resolución N.° 4067/MEGC/2021.

La propuesta de esta serie se enmarca en las Resoluciones N.° 321/MEGC/2015 y N.° 1189/MEGC/2015 y sus modificatorias N.° 1189/MEGC/2015 y 3510/MEGC/ 2015, en la Resolución CFE N.° 263/15 y en los Núcleos de Aprendizaje Prioritarios para Educación Digital, programación y robótica aprobados por el Consejo Federal de Educación mediante la Resolución N.° 343/18.

Además, responde a las características y las modalidades de trabajo pedagógico señaladas en el documento Orientaciones para la Organización Pedagógica e Institucional de la Educación Obligatoria, aprobado por la Resolución CFE N.° 93/09, que establece el propósito de fortalecer la organización y la propuesta educativa de las escuelas de nivel secundario de todo el país. A esta norma actualmente vigente, se agrega el documento MOA - Marco de Organización de los Aprendizajes para la Educación Obligatoria Argentina, aprobado por la Resolución CFE N.° 330/17, que plantea la necesidad de instalar distintos modos de apropiación de los saberes que den lugar a nuevas formas de enseñanza, de organización del trabajo docente y del uso de los recursos y los ambientes de aprendizaje.

En todas las normas mencionadas se promueven diversas modalidades de organización institucional, un uso flexible de los espacios y de los tiempos y nuevas formas de agrupamiento de las y los estudiantes, que se traduzcan en talleres, proyectos, articulación entre espacios curriculares, experiencias formativas y debates, entre otras actividades, en las que incluso participen estudiantes de diferentes años. En el ámbito de la Ciudad, el Diseño Curricular de la Nueva Escuela Secundaria incorpora temáticas emergentes y abre la puerta para el abordaje de problemáticas actuales de significatividad social y personal para la población joven.

La normativa vigente permite afirmar que existe acuerdo sobre la magnitud de los cambios que demanda el nivel secundario para lograr incluir al conjunto de estudiantes, y promover los aprendizajes necesarios para el ejercicio de una ciudadanía responsable y la participación activa en ámbitos laborales y de formación. En este sentido, si bien se ha recorrido un importante camino, es indispensable profundizar, extender e in-

corporar propuestas que ofrezcan reales oportunidades de aprendizaje y hagan de la escuela un lugar convocante y un espacio privilegiado para despertar inquietudes y vocaciones.

Los materiales que componen la serie articulan contenidos propios de los espacios curriculares de la formación general y de la formación específica de los bachilleratos orientados con contenidos de Educación Digital, Pensamiento computacional, Programación y Robótica. Ofrecen orientaciones y una guía de actividades que culminan con una producción que anticipa y plantea tres diferentes niveles de logro, de manera de contemplar los diversos contextos o entornos.

El común denominador de los materiales es proponer problemas y temáticas que resultan desafiantes e interesantes para los y las jóvenes que cursan la escuela secundaria y ofrecer oportunidades y estrategias para que "aprendan haciendo", diseñen, creen y recreen de manera sencilla y accesible productos y/o artefactos en forma individual o grupal con la guía del o de la docente. Al mismo tiempo, contribuyen al desarrollo gradual de capacidades para la exploración y el trabajo autónomo, a partir de las orientaciones precisas y claras sobre los procedimientos adecuados para el manejo de aplicaciones y de los entornos virtuales. Se espera que, a partir de estas experiencias, los y las estudiantes puedan apasionarse y continuar en forma individual o con sus compañeros y compañeras la indagación de otros problemas que conectan tecnología, ciencia, filosofía, sociedad, política y cultura.

Cabe aclarar que, en algunos casos, se podrá adoptar la propuesta completa, y, en otros, seleccionar las partes que se consideren más convenientes. Asimismo, se podrá plantear un trabajo de mayor articulación o exigencia de acuerdos entre docentes, puesto que serán los equipos de profesores y profesoras quienes podrán tomar las decisiones didácticas en las que el uso de estos materiales cobre sentido.

Confiamos en que estos recursos didácticos constituirán un gran aporte para el trabajo cotidiano en las instituciones educativas de nivel secundario y como toda serie en construcción, seguirá incorporando y poniendo a disposición de las escuelas de la Ciudad nuevas propuestas, que darán lugar a nuevas experiencias y nuevos aprendizajes.

Javier Simón
Director General
de Planeamiento Educativo

Eugenio Visiconde
Gerente Operativo
de Currículum

¿Cómo se navegan los textos de esta serie?

Los materiales de la serie Educación Digital, Programación y Robótica cuentan con elementos interactivos que permiten la lectura hipertextual y optimizan la navegación.

Itinerario de actividades

Actividad 1

Organizador interactivo que presenta la secuencia completa de actividades.

Para visualizar correctamente la interactividad se sugiere bajar el programa Adobe Acrobat Reader que constituye el estándar gratuito para ver e imprimir documentos PDF.

Pie de página

« 6 » Folio, con flechas interactivas que llevan a la página anterior y a la página posterior.

Volver a vista anterior Al cliquear regresa a la última página visitada.

índice interactivo

Introducción · Contenidos · Escenario · Actividades · Evaluación · Anexo

Al pie de cada página se encuentra el índice interactivo, que lleva a todas las secciones del documento.

Señalizadores gráficos

Estos iconos facilitan la localización de información relevante para el/la usuario/a, desde la columna lateral de la página.

ETIQUETAS

Palabras clave en el planteamiento del escenario y de las actividades.

Importante
Conceptos, recomendaciones, o reflexiones.

Archivos
Documentos para descargar.

Tutorial
Información

Tutoriales o instructivos.

Presentación
Entrevista
Tutorial

Contenido audiovisual.

Tarjeta

Uso de tarjetas.

Glosario

Búsqueda de palabras en la sección de glosario.

Introducción

En esta secuencia didáctica se propone resolver una necesidad relacionada con el almacenamiento, el procesamiento, la producción y la transmisión de información en formato digital a partir la creación de un **sistema informático** que permita sistematizar la carga de información y la obtención de resultados y conclusiones, aplicando técnicas informáticas de representación, organización y modelización de información de acuerdo con la situación planteada.

A través de la *metodología de enseñanza y de aprendizaje basada en proyectos*, se promueve abordar los procesos de diseño y creación de soluciones tecnológicas que no solo implican proyectos informáticos en sí, sino también proyectos que integran los desarrollos informáticos como componentes sustantivos, lo que estimula la creación de soluciones basadas en necesidades de la comunidad o en problemáticas concretas.

El eje que se considera en esta secuencia didáctica es el que corresponde al desarrollo de estrategias y técnicas relacionadas con la organización y con la gestión de los datos. Esto incluye contenidos relacionados con bases de datos, y abarca su gestión, diseño y creación.

En esta propuesta de enseñanza se desarrolla un sistema interactivo con interfaz gráfica, en el que se incorpora el almacenamiento en bases de datos —locales o en la nube— y el procesamiento de esos datos para la obtención de información o conclusiones de acuerdo con la problemática por atender. En este proceso, se evidenciarán la presencia del pensamiento computacional y la aplicación de la programación requerida para obtener el *software* como producto. A su vez, se habilita a quien lo desee a vincular el sistema con una aplicación móvil (en el caso de que el sistema incorpore bases de datos en la nube).

Como caso de trabajo, se abordarán los conceptos generales y se implementará un proyecto base realizado en el entorno de desarrollo Visual Studio con el lenguaje C# (ver Glosario), con integración de bases de datos locales (con la posibilidad de incluir bases de datos en la nube Firebase) y la generación de reportes. Se incorporan otras versiones de acuerdo con el nivel alcanzado. No se implementarán ni incorporarán contenidos curriculares vinculados con los conceptos de *arquitectura* (pueden averiguar más sobre Clean Arch) o *patrones de diseño* (pueden averiguar más sobre Model-View-ViewModel (MVVM)), con la intención de enfocar la construcción de conocimientos y habilidades básicas en la creación de proyectos de desarrollo de *software*.

Glosario

7

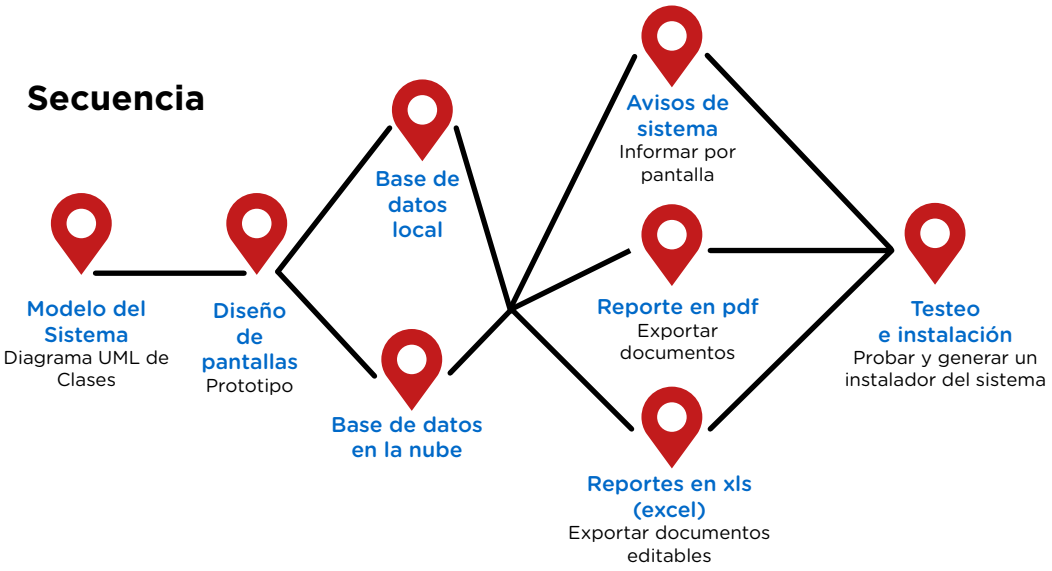
Introducción · Contenidos · Escenario · Actividades · Evaluación · Anexo

Volver a vista anterior

Los contenidos curriculares que se abordan en esta secuencia didáctica son los siguientes:

- Diseño y modelización general de un sistema informático.
- Entidades de un sistema. Diagramas UML.
- Casos de uso. Análisis funcional.
- Diseño y creación de las interfaces gráficas del sistema. Interacciones y menús del sistema. Respuesta a eventos.
- Diseño y creación de las bases de datos.
- Carga y procesamiento de datos.
- Salidas del sistema, qué información relevante presentamos a los/as usuarios/as y cómo.
- Testeo.

Crear soluciones: ¿cómo construir un sistema a medida?



Objetivos de aprendizaje, contenidos y capacidades

Objetivos de aprendizaje

Desde la orientación en Informática se propone:

- Resolver problemas del campo informático mediante el diseño de algoritmos y la aplicación de técnicas de representación apropiadas.
- Aplicar estrategias de programación seleccionando y combinando estructuras y utilizando los fundamentos básicos de la programación orientada a objetos.
- Desarrollar programas y aplicaciones que respondan a requerimientos específicos de uso y aplicación evaluando su eficiencia y su eficacia y elaborando la documentación técnica asociada.
- Experimentar con problemas y lenguajes de consultas de bases de datos relacionales identificando sus conceptos, técnicas y terminologías.
- Realizar proyectos de diseño de bases de datos construyendo diagramas entidad-relación y aplicando conceptos de normalización, redundancia y transacciones.
- Planificar y desarrollar productos o servicios informáticos reconociendo las etapas, asignando roles y tareas, estimando plazos, ejecutando el proyecto, generando la documentación técnica y definiendo criterios para la evaluación y ajuste del resultado obtenido.

Desde Educación Digital, Programación y Robótica se propone:

- Buscar, seleccionar, procesar, recuperar, sistematizar, jerarquizar, compartir e interpretar información disponible en múltiples formatos y soportes digitales, para transformarla en conocimiento.
- Crear con tecnologías digitales disponibles y, a la vez, ser capaces de crear nuevas tecnologías a partir del conocimiento de su funcionamiento y de los lenguajes que les son propios, con miradas críticas que permitan problematizarlas, discernir su utilidad, su potencial aplicación y sus implicaciones personales, sociales, locales y globales.
- Colaborar entre pares y trabajar en equipo, de forma cooperativa y colaborativa, para alcanzar un objetivo común, a través del acceso, el uso y la apropiación creativa de múltiples recursos digitales para distintos fines, de manera crítica, intencional y responsable, construyendo y participando en redes seguras de aprendizaje.

Contenidos	Capacidades
• El modelo computacional de objetos: características. Conceptos básicos: clase y objeto, atributos y métodos, estado y comportamiento, mensaje entre objetos, encapsulamiento de la información, tiempo de vida de los objetos. Abstracción y modularización. Herencia. • Lenguaje de modelado unificado (UML). Características y aplicaciones. • Diagramas de clases. Relaciones entre clases: herencia, asociación, composición y agregación. • Estándares de interfaces de usuarios/as. Principios generales. • Testeo. Concepto y objetivo. Documentación del testeo.	• Búsqueda, análisis y uso crítico de la información. • Trabajo en equipo y aprendizaje colaborativo. • Aprender a aprender. • Uso y creación de tecnologías digitales con una mirada crítica y creativa. • Resolución de problemas. • Comunicación.

Objetivos de aprendizaje	
Lógicas de programación • Análisis de situaciones complejas mediante el pensamiento computacional. Estudio de situaciones y planteo de soluciones a través de la abstracción, el reconocimiento de variables, la descomposición en situaciones más pequeñas, el reconocimiento de patrones y el desarrollo algorítmico. Bases de datos • Diseño, creación y modificación de bases de datos que permitan dinámicas que integren diversos desarrollos de *software* en el marco de un mismo proyecto.	

Escenario

En la actualidad se realizan grandes cantidades de transacciones y operaciones de forma digital. Pedidos, compras, reservas, trámites, pagos, registros y anotaciones son parte de las acciones de la vida cotidiana que han sido digitalizadas. La aparición de la telefonía celular y de mejores conexiones móviles y la disponibilidad de recursos tecnológicos han colaborado para que se produzca este proceso de transformación digital. Ahora bien, ¿todos los negocios, comercios y proyectos pueden o han podido digitalizarse? Esto depende de qué se esté analizando, pero seguramente existirá una parte o un proceso del proyecto que se puede ver beneficiado a partir de la digitalización. Esto permitirá analizar la información contenida y sacar conclusiones y estadísticas.

Se plantea vivenciar el proceso de tecnificar y digitalizar un proyecto, comenzando por la detección de una necesidad real de la comunidad, compatible con un modelo general que requiera el ingreso y carga de datos para luego procesarlos, presentarlos en pantalla y obtener una salida del sistema; por ejemplo, la generación de documentos en formato PDF, xls, txt, etcétera.

Para ello se propone construir un sistema de gestión desarrollado para el sistema operativo Windows. Así, los/as estudiantes desarrollarán habilidades y adquirirán conocimientos sobre el desarrollo de proyectos basados en .NET y codificados en lenguaje C#. Se espera que el proyecto incorpore bases de datos y el tratamiento de la información para dar respuesta a la problemática que lo motiva. De acuerdo con las decisiones que se quieran tomar, se presentarán las opciones en tarjetas. Cada tarjeta es una decisión de diseño distinta que compone el sistema. De acuerdo con las posibilidades de desarrollo, pueden incorporarse el uso de bases de datos de forma local como también en la nube, la generación o no de reportes, la carga de información de forma manual o sistematizada en una planilla xls, entre otras opciones.

Presentado el escenario, se describe el producto final con diferentes niveles de logro: básico, intermedio y avanzado.

Estos tres niveles de logro pueden ser abordados en función de las posibilidades y características de cada estudiante y cada grupo y del contexto.

ETIQUETAS

Abstracción

Descomposición

Modelización

Análisis e interpretación de los datos

Niveles de logro		
Nivel básico	**Nivel intermedio**	**Nivel avanzado**
Ejemplo: Aplicación ejecutable para PC que tome pocos datos por teclado, los almacene en una base de datos (ver Glosario) local, los procese y los presente en la interfaz gráfica. El sistema no exportará ningún reporte.	Ejemplo: Aplicación ejecutable para PC que tome datos de un archivo o por teclado, los almacene en una base de datos local, los procese y los presente en la interfaz gráfica. El sistema no exportará ningún reporte.	Ejemplo: Aplicación ejecutable para PC que tome datos de un archivo o por teclado, los almacene en una base de datos local, los procese y los presente en la interfaz gráfica. El sistema exportará diferentes reportes PDF, xls, txt, etc.

Itinerario de actividades

Actividad 1. Modelado y diseño del sistema

En esta actividad, cada grupo de estudiantes definirá una problemática para resolver y proyectará su solución informática, la que deberán diseñar, descomponer y modelar para su posterior desarrollo.

ETIQUETAS

Diseño de *software*
Abstracción
Modelado
Paradigmas de programación
Orientación a objetos

Actividad 2. Creación de interfaces gráficas

En esta actividad, y de acuerdo con el proyecto elegido, los/as estudiantes deberán determinar y diseñar las pantallas e interfaces gráficas del sistema.

ETIQUETAS

UI
UX
Interfaz gráfica
Diseño gráfico
Controles
Herramientas
Componentes
Interacciones

Actividad 3. Bases de datos

En esta actividad de aprendizaje los grupos deberán determinar qué datos se requieren, de qué tipo serán y cuál será el diseño de las tablas en donde estarán alojados para su posterior utilización.

ETIQUETAS

ABM
Datos
Base de datos relacional
Base de datos
SQL
Query

Actividad 4. Procesamiento de datos, reportes y testeo

En esta actividad los/as estudiantes tendrán que implementar la solución informática capaz de procesar datos para obtener una presentación de resultados. Deberán definir los reportes, estadísticas o conclusiones que se les presentan a los/as usuarios/as.

ETIQUETAS

Reportes

Generación de informes

PDF

Excel

Reporte editable

Actividad 5. Presentación y comunicación de proyecto y conclusiones

En esta actividad final los grupos deberán comunicar de forma efectiva lo elaborado, explicitando cómo fue el proceso de desarrollo y la solución obtenida.

ETIQUETAS

Presentación de proyecto

Elevator pitch

Ateneo de proyectos

Comunicación efectiva

Materiales necesarios

- Computadora con conexión a Internet.
- *Software* Visual Studio Community 2019.
- SQL Server 2019 Versión Desarrollador.
- Cuenta gratuita de Microsoft u Outlook.

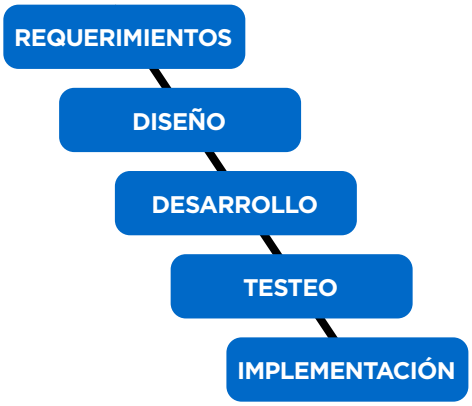
Tarjetas

- Tarjeta 1: Lectura de un archivo y carga en la base de datos.
- Tarjeta 2: Generación de reporte en archivo de texto.
- Tarjeta 3: Generación de reporte en archivo PDF.
- Tarjeta 4: Generación de reporte en archivo Excel.
- Tarjeta 5: Generación del instalador del sistema.

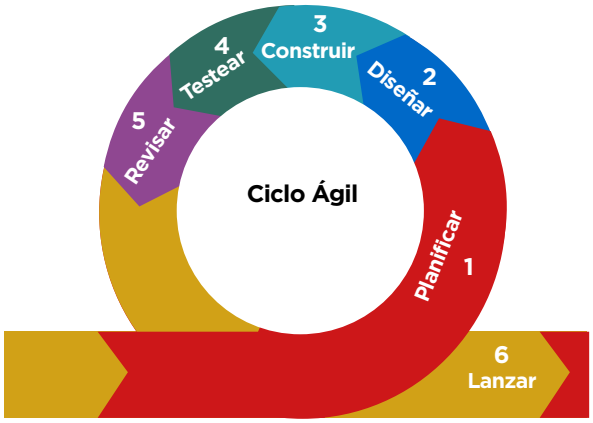
Tarjeta

Orientaciones para el desarrollo de las actividades

Actividad 1. Modelado y diseño del sistema



Metodología Cascada o WaterFall



Etapas del proceso de desarrollo según dos metodologías distintas.

Definir la idea-proyecto

Antes de comenzar a desarrollar un sistema, conviene pensar qué solución se requiere para una determinada necesidad. Por ejemplo:

Solución a una necesidad

- ¿Qué necesidad detecto en mi comunidad, escuela o negocio familiar?
- ¿Qué solución informática se me ocurre que pueda mejorar la tarea o dar respuesta a esa necesidad?

Funciones, objetivos y destinatarios/as

- ¿Qué función cumplirá mi sistema?
- ¿Cuál es su objetivo?
- ¿Quiénes serán los/as usuarios/as?

Aspectos funcionales

- Diseño:
  - ¿Cómo imagino que va a funcionar?
  - ¿Qué pantallas va a tener?
- Entrada de datos:
  - ¿De dónde va a provenir la información?
  - ¿Cuáles son mis fuentes de datos?
  - ¿Cómo ingresará la información el/la usuario/a?

- Procesamiento de información:
  - ¿Voy a procesar y a analizar la información?
  - ¿Dónde voy a guardarla?
- Salida de información:
  - ¿Se le mostrará algo a el/la usuario/a?
  - ¿Cómo se le presentará a el/la usuario/a?
  - ¿Obtendrá un reporte? ¿En qué formato de archivo?

En la vida cotidiana, interactuamos con muchos sistemas digitales; por ejemplo:

- Plataformas de películas y series a demanda (Netflix, Amazon Prime, Cine.ar, etcétera).
- Sistemas de videoconferencia (Meet, Zoom, WhatsApp, etcétera).
- Mensajería de texto por email.
- Registro en el calendario para recordar eventos o fechas importantes.

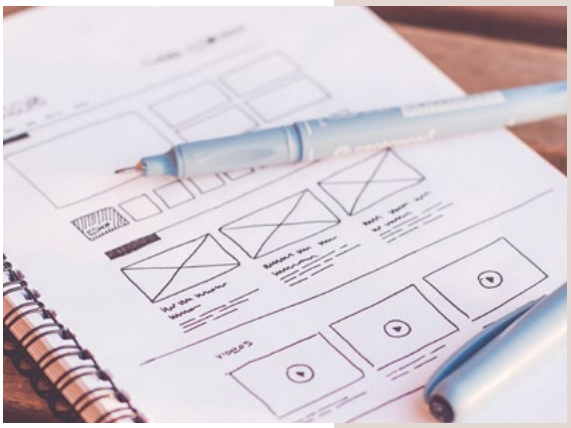
También es posible ser no solo usuarios/as de sistemas, sino creadores/as de uno propio. Podrían pensar en una solución para su comunidad, escuela o negocio familiar, para emprender un proyecto o para diseñar; por ejemplo:

- Un sistema de préstamos para la biblioteca de la escuela.
- Un sistema de préstamos de equipamiento de la escuela (materiales, computadoras, dispositivos, etcétera).
- Un sistema para una organización de la sociedad civil (ONG) o una institución de la comunidad que requiera digitalizar algún proceso o satisfacer una necesidad.

Ahora bien, para traducir la idea en una representación técnica del proyecto, los grupos de estudiantes deberán realizar diferentes diagramas y establecer la división de roles y responsabilidades:

- ¿Quién diseñará el sistema?
- ¿Quién asumirá el rol de programador/a? ¿Será uno/a solo/a o serán varios/as? En el caso de que sean varios/as, ¿Qué programará cada uno/a? ¿Cómo pueden establecerse acuerdos?
- ¿Quién será el/la tester de la aplicación? ¿Qué tipos de pruebas se realizarán?
- ¿Quién tomará el rol de líder de proyecto para organizar las tareas y tiempos de trabajo del equipo?

Prototipar la idea



Lo primero que se sugiere a la hora de pensar en un proyecto es bocetar la idea en pantallas para poner de manifiesto cómo sería y cómo van a interactuar los/as usuarios/as con las pantallas del sistema. Existen diferentes tipos de diseños de acuerdo con el nivel de detalle o de fidelidad que se desea tener. Se propone a los/as estudiantes que investiguen las diferencias entre:

- *Sketching* (ver Glosario)
- *Wireframe* (ver Glosario)
- *Mockups* (ver Glosario)
- *Prototype* (prototipo/maqueta) (ver Glosario)

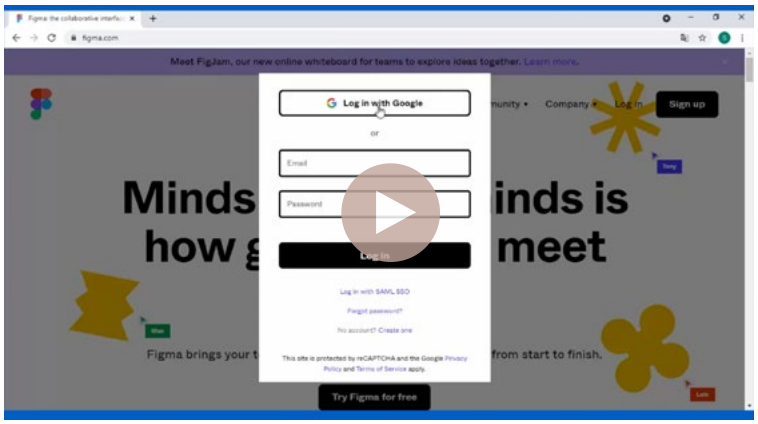
Glosario

Para realizar la investigación podrán tener en cuenta los siguientes criterios:

- Distinguir el **origen** de la información, descartando sitios irrelevantes o anuncios.
- Utilizar **palabras claves** para tener resultados más acertados con la temática de la búsqueda.
- Revisar los **dominios** para saber quiénes son los/as autores/as de la información.

Para ello se sugiere realizar un prototipo en el entorno de diseño Figma. Es un entorno de diseño y prototipado online y se pueden realizar diferentes tipos de representaciones y simulaciones:

a. Ingresen a Figma. Inicien sesión o realicen el registro con su cuenta. Pueden consultar el siguiente tutorial:



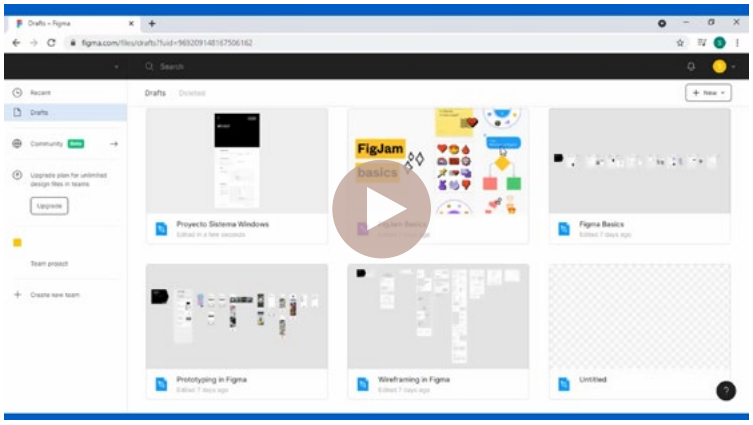
"Crear o iniciar sesión en Figma"

Tutorial

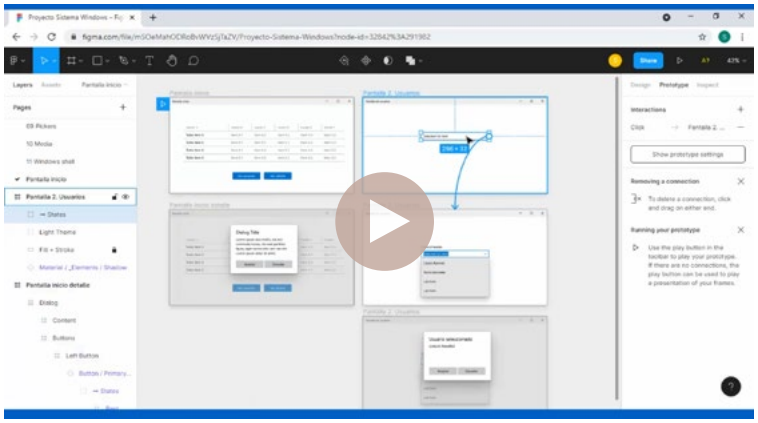
b. Descarguen la siguiente plantilla de componentes.

Archivos

c. Importen la plantilla para generar un proyecto nuevo. Para importar los componentes y para comenzar a utilizarlos y generar interacciones, se incluyen a continuación dos tutoriales:



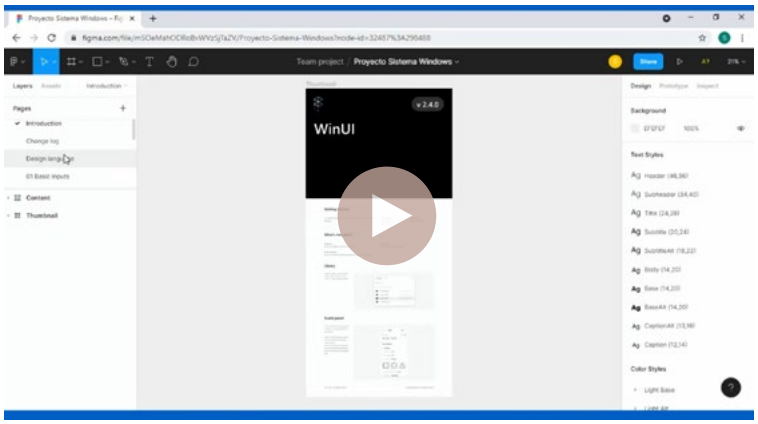
"¿Cómo importar plantilla en Figma?"



"¿Cómo usar componentes en Figma?"

d. Diseñen sus pantallas e interacciones teniendo en cuenta el objetivo del sistema y los aspectos principales de su usabilidad.

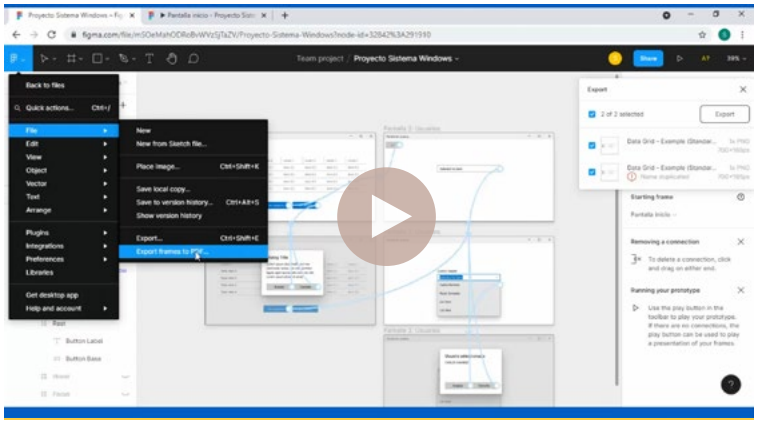
e. Conviertan el borrador en un proyecto y compártanlo con sus compañeros/as de equipo. Vean el siguiente tutorial, donde se explica el paso a paso para esta tarea:



"Compartir proyecto en figma"

Tutorial

f. Una vez que se hayan armado las pantallas, exportarán el prototipo realizado en PDF y también obtendrán el enlace. Para hacerlo, se puede ver el siguiente tutorial:



"Exportar diseño"

Una vez que está claro el/la destinatario/a y las interacciones que debe/puede realizar, será necesario adentrarse un poco más en qué información se requiere en cada pantalla.

Se compartirá el prototipo de sistema en un muro colaborativo elaborado, por ejemplo, con Jamboard o Miró, para permitir que otros grupos de estudiantes realicen comentarios y preguntas sobre las decisiones de diseño que se han tomado.

Definir el sistema

Los diagramas UML son un tipo de diagramas pensado para modelar las estructuras de datos y las relaciones presentes en un sistema de información. De la misma forma que un/una arquitecto/a dibuja y diseña planos sobre el edificio que va a construir, un/a analista de *software* (u otro perfil que cargue con este rol) crea distintos diagramas UML que sirven de base para el posterior desarrollo del sistema. El modelado es la principal forma de visualizar el diseño de una aplicación, con la finalidad de cotejarla con los requisitos antes de que el equipo de desarrollo comience a codificar.

Diagrama de estructura -> Representación de clases

A la hora de pensar cómo va a funcionar el sistema, conviene representarlo y proyectarlo. Para ello, una herramienta muy útil es la representación técnica mediante diagramas UML. El **Lenguaje de Modelado Unificado (UML) abstrae y representa aspectos de los sistemas**. El lenguaje de modelado es, por lo tanto, una herramienta práctica

Tutorial

para los/as desarrolladores/as de programas y sistemas. Dentro de los tipos de diagramas de estructura se encuentra el diagrama UML de representación de clases, donde cada una de ellas modela una entidad relevante del sistema de información.

¿Qué información se debe utilizar en el sistema? ¿Cómo se vinculan las diferentes partes del sistema?

- Dado que deberán programar un sistema cuyo paradigma está orientado a objetos, lo primordial es entender sus aspectos generales. Se observará entonces el video "Tutorial - Diagrama de Clases UML", en *Lucidchart Español*, donde se encuentra una explicación sobre los diagramas de clases.
- ¿Qué clases deben definir en su sistema? Como se observa en el video, el diagrama permite determinar qué instancias de clases (objetos) se van a utilizar y cómo se vinculan con otros objetos. Se elaborará el diagrama UML de clases de su sistema. Puede utilizarse, por ejemplo, en LucidChart o draw.io.

Tutorial

Diagrama de comportamiento -> Representación de casos de uso

A la hora de pensar las diferentes interacciones a las que debe dar respuesta el sistema, conviene representarlas y proyectarlas. Para ello, una herramienta muy útil es la representación técnica mediante diagramas UML de casos de uso.

Dentro de los tipos de diagramas de comportamiento se encuentra el diagrama UML de casos de uso, donde cada uno de esos casos representa una acción que puede realizar un/a usuario/a de nuestro sistema, configurando diferentes escenarios.

¿Mi sistema contempla diferentes tipos de usuarios/as? ¿Qué acciones puede realizar cada uno/a?

- Se observará el video "Elaboración de Diagrama de Casos de Uso con Lucidchart", en el canal de Claudia María Reyes Rangel, que ofrece una explicación sobre los diagramas de casos de uso.
- ¿Qué acciones deben definir en su sistema? Como se observa en el video, el diagrama permite determinar qué interacciones se deben contemplar. Se realizará el diagrama UML de casos de uso de su sistema. Podrá utilizarse, por ejemplo, en LucidChart o draw.io.

Tutorial

Ya está modelada y diseñada la idea del sistema. Ahora se procede a conocer cómo traducir esas ideas en un *software* que per-

Importante

mita que sus componentes e interacciones se implementen para satisfacer la necesidad inicial.

Conociendo el entorno de desarrollo integrado (IDE)

Para esta secuencia se utiliza el IDE Visual Studio en su versión Community 2019. Puede descargarse gratuitamente en "Visual Studio Community".

Para el proyecto utilizarán el IDE indicado y desarrollarán bajo una plataforma conocida como .NET (ver "A tour of the C# language". Se puede aplicar el traductor de *Google Translate* para poder leer el contenido en español), que es el estándar elaborado por Microsoft para el desarrollo de soluciones informáticas. En particular, utilizarán el lenguaje C#.

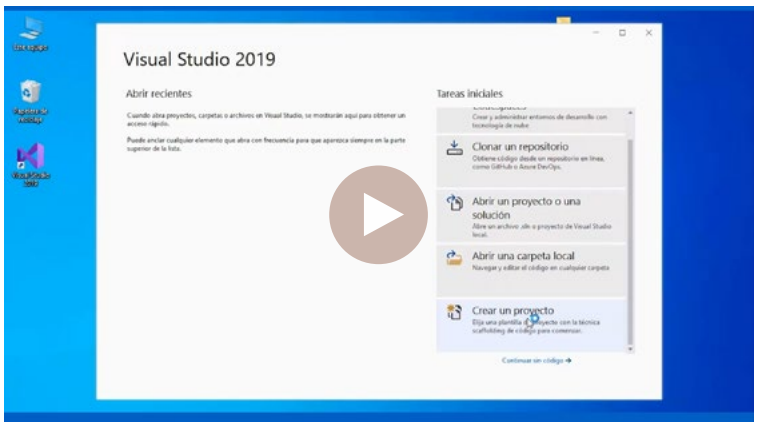
El tipo de proyecto será WPF, *Windows Presentation Foundation* (ver Glosario), una tecnología de Microsoft que permite el desarrollo de interfaces de interacción en Windows tomando características de aplicaciones de Windows Forms y de aplicaciones web.

Glosario

Lenguaje de programación C#

El C# es un lenguaje de programación que permite a los/as desarrolladores/as crear muchos tipos de aplicaciones seguras y sólidas que se ejecutan en el ecosistema .NET. C# tiene sus raíces en la familia de lenguajes C (ver "A tour of the C# language". Se puede aplicar el traductor de Google Translate para poder leer el contenido en español).

- Observen el siguiente video, en el que se explica el entorno de desarrollo Visual Studio Community 2019 con los pasos requeridos para comenzar un proyecto nuevo. ¡Manos a la obra!



"Instalar Visual Studio Community 2019".

Tutorial

Actividad 2. Creación de interfaces gráficas

Parte A. Conociendo los controles y herramientas disponibles para pantallas

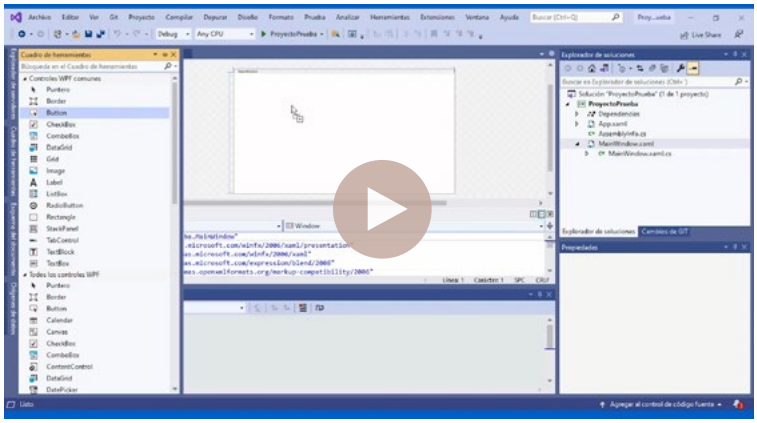
Para pensar en las pantallas, es preciso conocer primero los diferentes controles y elementos componentes de una interfaz gráfica. Dado que se ha planteado un desarrollo con Visual Studio, se conocerán diferentes controles que se pueden incorporar en sus interfaces gráficas o ventanas.

Todos los componentes se agregan arrastrándolos dentro de la ventana (la interfaz gráfica). Cada componente cuenta con dos espacios de configuración: "propiedades" y "controladores de eventos".

En la sección "propiedades" podrán configurar aspectos de diseño, como apariencia y características del componente, por ejemplo: alto, ancho, color de fondo, alineación, etcétera.

En la sección "controladores de eventos" podrán registrar la respuesta al evento deseado, por ejemplo: responder ante un clic sobre el componente, doble clic, arrastre, posicionamiento del *mouse*, etcétera.

- Observen el siguiente video, donde se presentan las diferentes opciones:



"Conociendo el cuadro de herramientas".

- Se exploran las características generales, similitudes y diferencias entre controles como TextBox, Label, Button, ListBox, ComboBox, Checkbox, RadioButton.

Parte B. El diseño centrado en el/a usuario/a

La usabilidad y la experiencia de los/as usuarios/as son aspectos centrales de un sistema. Los/as estudiantes deberán tener en cuenta qué pasa si una ventana del sistema se maximiza o se redimensiona. ¿Qué pasa con los controles y las vistas? ¿Se puede ver e interactuar con el contenido?

Es preciso, entonces:

- Agregar, dentro de su primera ventana, un Textbox y un botón debajo. Ejecutar su proyecto. ¿Qué pasa si se redimensiona la ventana haciéndola muy grande o muy chica?
- Ajustar y acomodar el contenido utilizando diferentes layouts. Investiguen sobre los componentes Grid, StackPanel, WrapPanel, DockPanel, ScrollViewer, Canvas.

Importante

En los siguientes enlaces pueden verse ejemplos de implementación de cada uno de estos componentes:

- Layout Grid
- Layout StackPanel
- Layout WrapPanel
- Layout DockPanel
- Layout ScrollViewer
- Layout Canvas

- Comparar en el propio muro colaborativo las características de cada uno de estos elementos ordenadores de contenido.

Parte C. Diseñando las pantallas de mi proyecto

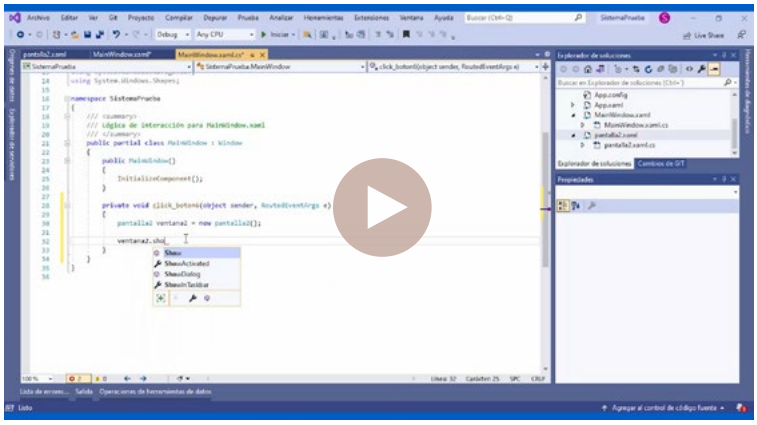
Un prototipo es un modelo gráfico (idealmente, interactivo) que permite representar las pantallas y las interacciones que suceden en nuestro sistema.

Importante

Ya han realizado el prototipo en Figma. A continuación, los/as estudiantes deberán replicar el diseño, pero en las interfaces gráficas del sistema y utilizando diferentes controles.

La pantalla que figura como “MainWindow” corresponde a la pantalla principal de su sistema.

- Se verá ahora el siguiente video sobre el paso a paso para crear una segunda pantalla en su proyecto:



“Crear otra pantalla”

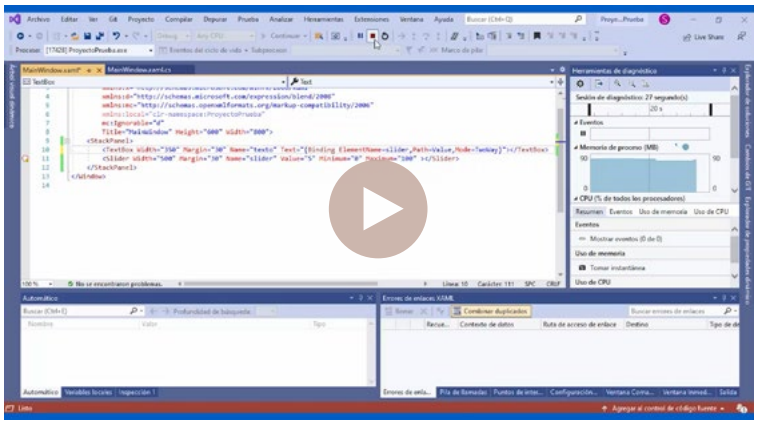
- Una vez que se han creado las diferentes pantallas, se colocan en cada una de ellas los controles necesarios desde la sección “Cuadro de herramientas”.

Parte D. Implementando interacciones entre diferentes controles y datos

Se ha visto que se puede programar respuestas a los eventos de los diferentes controles, pero... ¿es posible vincular los datos que se encuentran en las pantallas? ¿Cómo hacer que un cambio de datos de un control produzca un cambio en otro componente? ¿Qué técnicas de vinculación de componentes se pueden implementar para presentar datos?

Data Binding

El enlace de datos (*Data Binding*) es el proceso que establece una conexión entre la interfaz de usuario/a de la aplicación y los datos que muestra. Si el enlace tiene la configuración correcta y los datos proporcionan las notificaciones adecuadas, cuando los datos cambian su valor los elementos que están vinculados a los datos reflejan los cambios automáticamente. El enlace de datos es de dos tipos: enlace de datos unidireccional y enlace de datos bidireccional. Se recomienda ver el siguiente video:



"DataBinding con TextBox + Slider"

Interface INotifyPropertyChanged

Las interfaces son una abstracción estupenda que la mayor parte de los lenguajes de programación orientados a objetos ofrece. Básicamente, permiten definir un "contrato", sobre el que se puede estar seguro de que las clases que las implementen lo van a cumplir. Por ejemplo, sobre la implementación de la interfaz INotifyPropertyChanged entre tres TextBox, se puede ver el video "InterfazINotifyPropertyChanged".

Parte E. Haciendo uso de diferentes controles

En este apartado se verán otros controles que se pueden utilizar para presentar información y realizar acciones.

Para esta parte de la actividad, se recomienda ver los siguientes tutoriales:

- "Uso de un ListBox".
- "Uso de un ComboBox".
- "Uso de Checkbox".
- "Uso de RadioButtons".

Tutorial

Actividad 3. Bases de datos

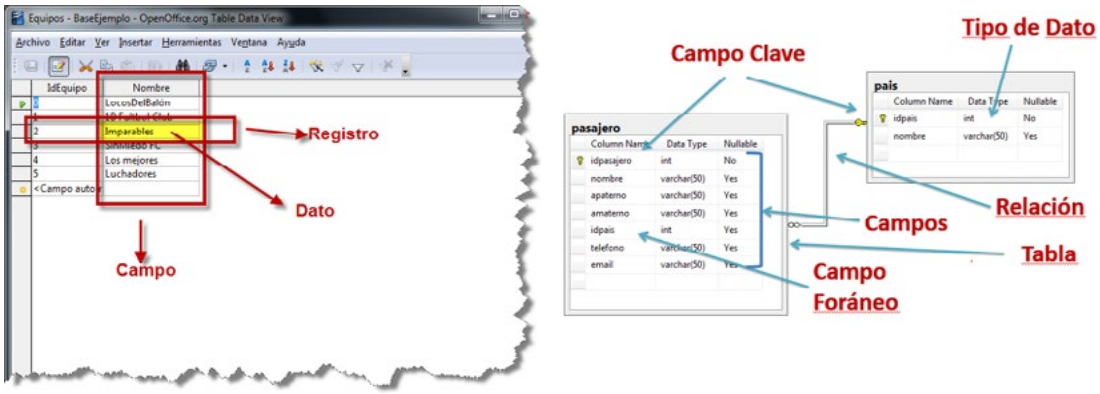
Se ha mencionado en actividades anteriores la importancia de la representación y de la modelización de los datos.

> Ahora que ya se dispone de las pantallas, los/as estudiantes deberán seguir con el diseño de las bases de datos (ver Glosario) donde se guardará la información.

Importante

Toda base de datos relacional se organiza en tablas (ver Glosario). Cada tabla cuenta con registros (ver Glosario) y cada registro con campos (ver Glosario). De todos ellos deberá existir, al menos, un campo primario que permita identificar y diferenciar un registro de información de otro.

Glosario

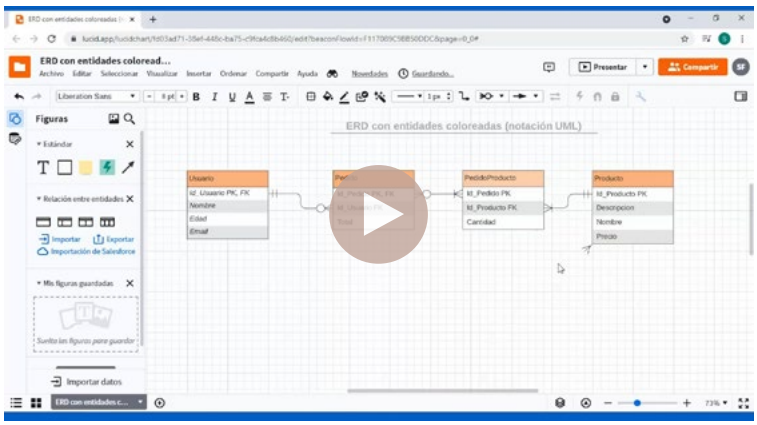


a. Vean el video "Tutorial - Diagrama Entidad-Relación (ER)", en *Lucidchart Español*, sobre las bases de datos y su representación mediante el diagrama de entidad-relación.

Tutorial

b. Vean el video "Tutorial - Diagrama Entidad-Relación (ER) Parte 2", en *Lucidchart Español*, sobre la "clave primaria" de una tabla de bases de datos relacional.

- ¿Cómo se relacionan diferentes tablas en una base de datos relacional? ¿Cuáles son las diferencias entre una clave primaria (PK) y una clave foránea (FK)?
- Es momento de armar su diagrama de entidad-relación. Puede realizarse en papel, en LucidChart, en draw.io o en el soporte que se elija. En el siguiente video puede verse un ejemplo:



"Armar un diagrama ERD con Lucidchart"

- Seguidamente, los/as estudiantes responderán las preguntas en su muro colaborativo.

c. Investigarán sobre los siguientes aspectos y dimensiones que tienen que ver con las bases de datos:

- ¿Qué es un SGBD o DBMS?
- ¿Qué son la redundancia, la integridad y la normalización de datos en una base de datos?
- ¿Qué representa cada nivel de normalización?
- ¿Qué tipos de bases de datos existen?
- ¿Qué representa la sigla CRUD o ABM en el uso de una base de datos?

Enlaces recomendados:

- ¡Base de datos! Sitio donde se puede encontrar información.
- "¿Qué es una base de datos?", en *Oracle*.

- Adicionalmente, agregarán la información en su muro colaborativo.

d. Para utilizar una base de datos SQL, deberán instalar un gestor de bases de datos. Este gestor permite interactuar con la base de datos y sus tablas. Para comenzar, se instala SQL Server 2019 Developer / Desarrollador, que se puede descargar aquí. Puede verse el anexo 3 "Instructivo de instalación de SQL Server 2019 - Versión Desarrollador".

Archivos

En los siguientes tutoriales se presentan los pasos sugeridos para implementar una base de datos relacional en C#:

- "Creación y conexión a una base de datos SQL".
- "Crear tablas de la base de datos".
- "Implementando consultas SQL".

Las consultas SQL permiten el alta, la baja y la modificación de datos en sus tablas de las bases de datos. En "¿Qué es una consulta en base de datos?", en *Hostinger Tutoriales*, pueden observarse las diferentes consultas simples que se utilizan generalmente.

Tutorial

Es conveniente observar los siguientes tutoriales para comenzar a implementar operaciones con bases de datos:

- "Mostrar datos en la interfaz gráfica".
- "Presentando datos en un datagrid".
- "Borrar registros de la tabla (Operación Baja o B)".
- "Manejo de errores y excepciones".
- "Agregar registros a la tabla (Operación Alta o A)".
- "Modificar registros de las tablas".

Sobre la base de su idea de proyecto, los/as estudiantes diseñan las tablas de base de datos requeridas e implementan las operaciones necesarias. La información por almacenar puede provenir del ingreso por teclado o a través de la lectura de un archivo. Para ver cómo importar o leer información desde un archivo csv, puede recurrirse a la tarjeta 1: Lectura de un archivo y carga en la base de datos.

Tarjeta

Actividad 4. Procesamiento de datos, reportes y testeo

Habitualmente, cuando un/a usuario/a administra o gestiona un sistema, suele requerir alguna forma de saber el estado de situación general o de alguna cuestión particular de ese sistema que resulta relevante. Por ejemplo, un sistema de préstamos debería poder tener una pantalla que señale primero aquellos préstamos cerca de la fecha y hora de finalización y que todavía no han sido devueltos. También,

aparte de verlo en una pantalla, pueden solicitar el listado de los préstamos realizados en un día en particular o conocer el listado de los préstamos que deberían devolverse hoy. Allí cobra particular relevancia la posibilidad de generar documentos de reporte o informes.

A la hora de recorrer y procesar el contenido de una tabla con información o control con contenido, puede recurrirse a estructuras de control. Entre las estructuras de control de flujo más básicas podrán encontrar, por ejemplo, las siguientes:

- ***If.*** Estructura condicional de control de flujo que permite definir cómo continúa la ejecución de acuerdo con su condición. Si es verdadera, toma un camino de ejecución, mientras que si resulta falsa toma otro (ver "Estructuras condicionales simples y compuestas", en *Tutoriales Programación Ya*).
- ***While.*** Estructura repetitiva de control de flujo, donde el bloque de código que se encuentra dentro del *While*, se repite mientras la condición sea verdadera (ver "Estructura repetitiva while", en el mismo sitio de tutoriales).
- ***For.*** Estructura repetitiva de control de flujo, donde el bloque de código que se encuentra dentro del *For* se repite mientras la condición sea verdadera. Esto resulta equivalente al *While*, con la diferencia de que la condición está asociada al incremento/decremento de un índice, que determina la cantidad de veces que debe repetirse el bloque de código (ver "Estructura repetitiva for", en el mismo sitio de tutoriales).

- A su vez, pueden utilizar una estructura de control de acuerdo con el contenido a evaluar con múltiples opciones. La estructura condicional *Switch* permite evaluar una variable o atributo y seleccionar el camino de respuesta de acuerdo con su valor entre múltiples opciones (ver "Estructura condicional switch", en *Tutoriales Programación Ya*).
- Por último, y para procesar o recorrer colecciones de datos, se utiliza una estructura repetitiva o iterativa llamada *Foreach*, similar a la estructura *For* pero pensada para un conjunto de datos determinado, barriendo secuencialmente todo el conjunto (ver "Estructura repetitiva foreach", en el mismo sitio de tutoriales).

Puede resultar particularmente interesante incluir alguna evaluación condicional de un campo de un registro, para ver si cumple cierta condición para ser parte de un reporte (por ejemplo, se quiere generar un reporte solo de aquellos registros cuyo campo "fecha" es igual a la actual, o cuyo campo "cantidad" es mayor a 10, o cuyo precio es mayor a 100, etcétera). En ese caso, podría involucrarse una estructura *if*

Tutorial

dentro del *Foreach* para aplicar el filtro deseado, o también es posible aplicar el filtro directamente desde la consulta SQL para obtener los registros que cumplen esa condición (ver "Consulta SQL con filtro de datos", en *Línea de código. Aprender a programar*).

De acuerdo con lo planteado como casos de uso del sistema, se definirá en qué formato y soporte se presentará la información requerida:

- ¿Será presentada en un reporte o archivo generado? ¿Lo presentarán en PDF, txt, Excel?
- ¿Será presentada en la interfaz gráfica del sistema? ¿Se priorizará la información requerida por sobre la estética? ¿Podrá el/la usuario/a, a partir de ello, exportar un listado?

- En la tarjeta 2: Generación de reporte en archivo de texto puede verse el paso a paso para generar un reporte en formato txt.
- En la tarjeta 3: Generación de reporte en archivo PDF puede verse el paso a paso para generar un reporte en formato PDF.
- En la tarjeta 4: Generación de reporte en archivo Excel puede verse el paso a paso para generar un reporte en formato xls (Excel).

Tarjeta

Testeo

Una parte fundamental en el proceso de desarrollo de un sistema, previamente a su implementación, es realizar el testeo de cada caso de uso contemplado. Es una actividad más en el proceso de control de calidad del sistema.

Para ello, se sugiere desarrollar un informe de *testing* (ver Glosario). El informe contiene información sobre el registro de las discrepancias entre el resultado esperado y el resultado obtenido en la ejecución de la funcionalidad del *software*. Esas discrepancias se pueden identificar en diferentes etapas del ciclo de vida del desarrollo del *software*. Con el objetivo de minimizar diferentes tipos de riesgos asociados a

Glosario

los proyectos y al producto obtenido, lo ideal es evidenciar la mayor cantidad de defectos posible en etapas tempranas del desarrollo del *software*, incluso desde la fase de planificación.

El informe debe contener los siguientes elementos:

Datos generales

- **Datos del proyecto** que ha sido objeto de las verificaciones.
- **Responsable del reporte**, quien encuentra el defecto y redacta el reporte en su rol de tester.

Por cada incidencia encontrada

- Especificar en qué **versión del proyecto**, funcionalidad, producto, se encontró el defecto.
- **Ambiente:** documentar si se utiliza la base de datos de producción (se asume cuando existe una sola) o de pruebas (si existe un doble ambiente, para desarrollar y para implementar), el nombre o el identificador del ambiente.
- **Resumen:** escribir un título que permita interpretar fácilmente el defecto.
- **Gravedad:** la gravedad facilita la priorización para la resolución de la incidencia y afecta los datos que se obtendrán en el informe.
- **Descripción del defecto**:
  - **Cuándo:** acción que describe el evento y las variables dentro de la descripción del caso de prueba.
  - **Qué:** resultado que se obtuvo al ejecutar la acción (cuándo), que discrepa del resultado esperado.
  - **Dónde:** ubicación del objeto de prueba.
  - **Resultado esperado:** describe el objetivo de la funcionalidad.
- **Pasos para reproducir el error:** describir todos los pasos que faciliten el camino para que cualquier persona del equipo pueda llegar a reproducir el defecto.
- **Información adicional:** datos utilizados en la prueba.
- **Reproducibilidad:** identificar si es un defecto común, de carácter "aleatorio", o si es un defecto frecuente, de forma de identificar la causa que lo origina.
- **Imágenes y/o videos complementarios:** respaldar el reporte con imágenes o videos para ampliar y facilitar su lectura.

Es conveniente dar el espacio para que otros/as integrantes del equipo o del curso prueben los sistemas para obtener *feedback*/retroalimentación que ayude a mejorar cada sistema.

Ya se ha desarrollado todo el sistema, ahora resta lograr que otros/as usuarios/as puedan probarlo. Resulta imprescindible que, en esta etapa de validación y retroalimentación, dejen a disposición el proyecto para que otros/as usuarios/as puedan ejecutarlo. Para eso se dispone del proyecto ejecutable en la subcarpeta bin del proyecto.

Importante

Opcional: Generar un instalador del proyecto

En la tarjeta 5: Generación del instalador del sistema puede verse el paso a paso para generar el instalador del proyecto. En el muro colaborativo se indica el enlace de Drive donde se pueda encontrar su instalador del *software*.

Tarjeta

Actividad 5. Comunicación y presentación de proyecto y conclusiones

Ya elaborado el sistema, desde la ideación al desarrollo y el testeo, llega el momento de presentarlo.

Se propone la siguiente situación imaginaria:

Imagínense que surge la oportunidad de conversar unos minutos con alguien importante, capaz de ayudarlos/as a hacer realidad su proyecto. ¿Qué le dirían? ¿Qué es importante contar en una presentación corta? ¿Hablarían de aspectos técnicos, funcionales o de la necesidad y la solución planteada?

Importante

Presentación tipo elevator pitch

Vean el video "Elevator pitch. Tienes 20 segundos – eduCaixa", en *iurisdocTV* y elaboren su propio elevator *pitch* (un minuto de duración).

Tutorial

Presentación pitch general

Vean el video "Cómo Hacer Un Pitch", en *BenjaminCoxMI*, con la explicación general de cómo plantear la presentación, y preparen la presentación PowerPoint de soporte. Desarrollen también una presentación de cinco minutos de duración.

Por último, se propone que los/as estudiantes dejen a disposición de los/as compañeros/as un formulario que sirva de retroalimentación para que respondan sobre los siguientes puntos:

- Qué han entendido del proyecto.
- Qué cosas no se han entendido.
- Qué consideran que es lo mejor del proyecto.
- Qué aspectos encuentran que se deben mejorar.
- Qué cosas no les han gustado.
- Dejar espacio para que formulen nuevas consultas o preguntas al equipo.

En el Campus Virtual de Educación Digital encontrarán un "Tutorial de Formularios de Google" para ver cómo preparar su formulario de retroalimentación. Dejen el enlace dentro del muro colaborativo, para responder.

Tutorial

Orientaciones generales para la enseñanza y la evaluación

Esta secuencia didáctica se centra en los siguientes bloques de contenidos curriculares:

- Lógicas de programación.
- Bases de datos.
- Informática y producción.

Los/as estudiantes podrán desarrollar capacidades y adquirir conocimientos vinculados a los siguientes temas:

- Algoritmos y estructuras de datos.
- Programación orientada a objetos.
- Desarrollo de aplicaciones.
- Diseño y creación de bases de datos.
- Gestión de bases de datos.
- Proyectos informáticos.

Respecto de la enseñanza, se ofrecen aquí las siguientes recomendaciones:

- Brindar espacios para el desarrollo del pensamiento de diseño, estimulando la creatividad y la colaboración entre pares.
- Se sugiere habilitar a que diseñen en papel y realicen un boceto a mano alzada de lo que pretenden que sean las interfaces gráficas del sistema, a modo de wireframe o prototipo simplificado.
- Dado que la metodología elegida es la enseñanza basada en proyectos, podrá realizarse la evaluación paso a paso a medida que se avanza a través de las etapas.
- Se sugiere utilizar el muro colaborativo y la etapa de presentación de proyectos como espacios para la comunicación de lo realizado y para evidenciar los aprendizajes y la toma de decisiones realizada.
- Se sugiere orientar la evaluación en función de la validación de aprendizajes vinculados a la toma de decisiones en el desarrollo del proyecto, la división de roles y tareas, la claridad conceptual con la que comunican las decisiones de diseño, implementación y testeo realizadas a lo largo del proceso.

Es posible plantear la implementación de una lista de cotejo para la evaluación del trabajo por etapas como la que sigue, para que los/as estudiantes, en conjunto con su docente, vayan paulatinamente dejando registros del trabajo individual y grupal que se va realizando, indicando la fecha en cada caso.

Niveles de logro					
Etapa	**Trabajo individual (Comentario de los/as estudiantes)**	**Trabajo grupal (Comentario de los/as estudiantes)**	**Logros alcanzados por el grupo (Comentarios del/de la docente)**	**Para ajustar o rehacer (Comentarios del/de la docente)**	**Fecha**
1. Etapa de definición de la idea					
2. Etapa de diseño					
3. Etapa de desarrollo					
4. Etapa de pruebas					
5. Etapa de comunicación y presentación					
6. Etapa de reflexión sobre lo realizado					
Sobre la totalidad del proyecto					

Antes de dar por cerrado el proyecto, puede implementarse en forma oral la rutina de pensamiento[1] titulada "Antes pensaba... ahora pienso...", y dejar constancia en una pizarra física o virtual (Jamboard, Padlet o equivalente), siguiendo el siguiente formato para la reflexión individual:

Antes pensaba...	Ahora pienso...

1 Una rutina de pensamiento es un conjunto de preguntas o una breve secuencia de pasos que se siguen para andamiar los procesos de reflexión de los/as estudiantes.

El formato que sigue —similar al anterior— apunta a la reflexión grupal, ya sea de los pequeños grupos como del grupo-clase total:

Antes pensábamos...	Ahora pensamos...

Se trata de una rutina de pensamiento interesante para poner en palabras, para hacer visibles las reflexiones respecto de lo que ha sucedido en el tránsito a través del proyecto, identificando cambios y continuidades, saber cómo se han producido y por qué y establecer diferencias entre cuestiones individuales y cuestiones referidas a los grupos, tomando en consideración sus sensaciones, aprendizajes y habilidades adquiridos en el proceso.

Enlaces recomendados:

- "Antes pensaba... Ahora pienso..." (rutina de pensamiento). Proyecto Cero, Escuela de Graduados en Educación de la Universidad de Harvard.

Se deja aquí, para consulta, la carpeta "Proyecto de implementación de bases de datos local (ejemplo)", que fuera insumo para la producción de los videos.

Importante

Explorando fronteras

Para ir más allá, se propone explorar tres aspectos que cobran relevancia en los proyectos de *software* que se desarrollan a continuación.

Análisis de datos

Uno de los aspectos principales de los sistemas de información es determinar claramente qué datos necesitan ser recopilados para alcanzar el objetivo planteado. Muchas veces, los proyectos en sí y sus objetivos pueden resultar complejos, pero si la estrategia para recopilar los datos y para procesarlos es clara, esos objetivos pueden alcanzarse en etapas progresivas o en diferentes versiones.

Se sugiere escuchar, a continuación, la entrevista a Alan Elkin Minuchin, líder de equipo de Investigación y Desarrollo de la empresa ZoomAgri.

Zoom Agri busca innovar tecnológicamente en las cadenas agroindustriales a través del Procesamiento de Imágenes e Inteligencia Artificial, digitalizando el testeo, la inspección y la certificación de commodities agrícolas y alimentos. Puede leerse un poco más en "Zoomagri, la empresa que revolucionó la industria cervecera", en AgroEmpresario.

Entrevista a Alan Elkin Minuchin.

Entrevista

Uso de bases de datos en la nube

Los sistemas modernos que nos permiten realizar desde el celular la mayoría de las acciones de la vida cotidiana utilizan bases de datos en la nube. Eso significa que generan nuevos datos y consumen otros desde un repositorio disponible a través internet. Lo mismo sucede si se quiere conectar el sistema con una *app* móvil.

Uno de los entornos más utilizados es Firebase. Podrán ver el video "Presentando Firebase", en el canal del mismo entorno.

El canal *Fauxcode* contiene una serie de videos en los que podrán ver cómo realizar un ABM en su sistema WPF con bases de datos en la nube de Firebase. El primero de ellos es "C# Application to Firebase Database | Connecting to Firebase using Firesharp C# Application".

Material audiovisual

Aprender más sobre arquitectura y patrones de diseño MVVM

A la hora de construir un sistema sólido y escalable deben tenerse en cuenta aspectos vinculados a la arquitectura de la solución.

Una de las arquitecturas más importantes se denomina arquitectura *clean* o limpia (ver video "Arquitectura limpia!! o en inglés Clean Architecture!! En 5 minutos!! Fácil de entender", en *weincode*). Propone dividir a una solución de *software* en capas donde cada una se ocupa de un aspecto determinado, dividiendo responsabilidades y cumpliendo con los principios "Solid". En el video "Arquitectura de software en capas - Introducción y ejemplos con C# con inversión de dependencia", de Nicolás Battaglia, es posible conocer más sobre un ejemplo de implementación de arquitectura *clean* en un proyecto de *software* elaborado en C#.

Material audiovisual

Por otro lado, si bien la arquitectura *clean* tiene su subdivisión en capas, las interfaces gráficas involucran componentes e interacciones. Allí entran en juego patrones de diseño, por ejemplo, el patrón MVVM (Model-View-ViewModel). En el canal *Leodev*, se encuentra una serie de videos con la explicación práctica de su implementación, el primero de los cuales es "C# - Serie MVVM (Parte N°1)".

Anexos

Anexo 1. Tarjetas

Anexo 2. Glosario

Anexo 3. Instructivo de instalación de SQL Server 2019 - Versión Desarrollador

Anexo 1

Tarjetas

Link para descargar tarjeta imprimible

Tarjeta 1: Lectura de un archivo y carga en la base de datos

Paso a paso para leer un archivo de texto separado por comas (.csv) y su posterior carga en una base de datos.

- Para saber más sobre un archivo csv, pueden ingresar a "¿Qué es un archivo csv y para qué sirve?", en *Geeknetic*.
- En el video "Importar/leer desde un archivo de texto" podrán observar los pasos requeridos para poder leer una archivo de texto y cargar su información en una tabla de la base de datos. Pueden también acceder al video a través del siguiente QR:

Link para descargar tarjeta imprimible

Tarjeta 2: Generación de reporte en archivo de texto

Paso a paso para generar un reporte o documento en formato txt.

- En el artículo "Archivo de texto", en Wikipedia, podrán acceder a información sobre este tipo de archivo.
- En el video "Generar archivo de texto" podrán observar los pasos requeridos para generar un reporte o documento en formato txt. Pueden también acceder al video a través del siguiente QR:

Link para descargar tarjeta imprimible

Tarjeta 3: Generación de reporte en archivo PDF

Paso a paso para generar un reporte o documento en formato PDF.

- En "PDF: tres letras que continúan cambiando el mundo", en *Adobe Acrobat*, encontrarán información sobre este tipo de documentos.
- En el video "Generar reporte PDF" podrán observar los pasos requeridos para generar un reporte o documento en formato PDF. Pueden también acceder al video a través del siguiente QR:

Link para descargar tarjeta imprimible

Tarjeta 4: Generación de reporte en archivo Excel

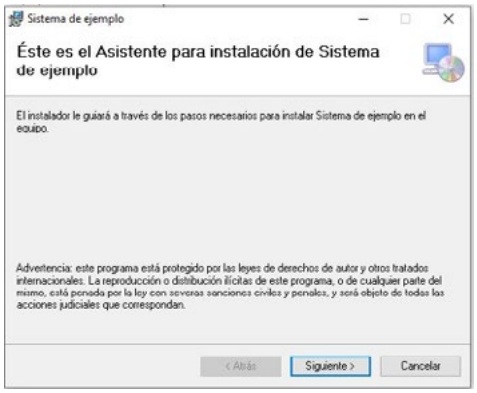
Paso a paso para generar un reporte en Excel.

- En "Archivos XLSX: ¿Qué son? ¿Cómo abrirlos?", en *tecnología+informática*, podrán ver un poco más sobre los documentos xlsx o planillas de cálculo Excel.
- En el video "Generar reporte de Excel" podrán observar los pasos requeridos para generar un reporte o documento en formato xlsx. Pueden también acceder al video a través del siguiente QR:

Link para descargar tarjeta imprimible

Tarjeta 5: Generación del instalador del sistema

Paso a paso para crear un instalador para su sistema.



- En esta oportunidad podrán ver un poco más sobre los instaladores de sistemas.
- Como primer paso, deberán descargar e instalar la extensión Visual Studio Installer Projects.
- En el video "Generar instalador del proyecto" podrán observar los pasos requeridos para poder generar el instalador. Pueden también acceder al video a través del siguiente QR:

Anexo 2

Glosario

Base de datos: es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. En este sentido, una biblioteca puede considerarse una base de datos compuesta en su mayoría por documentos y textos impresos en papel e indexados para su consulta.

C#: es un lenguaje de programación multiparadigma desarrollado y estandarizado por la empresa Microsoft como parte de su plataforma .NET.

Campo: es la unidad básica de entrada de datos de un registro. Corresponde al nombre de la columna. Debe ser único y además de tener un tipo de dato asociado.

Diagramas UML: son un tipo de diagramas pensado para modelar las estructuras de datos y las relaciones presentes en un sistema de información. De la misma forma que un/una arquitecto/a dibuja y diseña planos sobre el edificio que va a construir, un/a analista de *software* (u otro perfil que cargue con este rol) crea distintos diagramas UML que sirven de base para el posterior desarrollo del sistema

Mockup: es una maqueta estática representando el diseño del sistema. Se incorporan colores, tipografías y piezas gráficas.

Prototipo: es una construcción interactiva que permite simular el comportamiento del sistema, incorporando colores, tipografías, piezas gráficas e interacciones.

Registro: también llamado *fila* o *tupla*, representa un objeto único de datos implícitamente estructurados en una tabla. En términos simples, una tabla de una base de datos puede imaginarse formada por filas (registros) y columnas (campos o atributos).

Sketch: generalmente, hace referencia a un boceto. Es un primer apunte de un diseño de producto que se llevará a cabo más adelante de manera más detallada.

Tabla: es la organización de datos dentro de una base de datos. Se refiere al tipo de modelado de datos donde se guardan los datos en una estructura compuesta de registros y cada uno de ellos representados en diferentes campos.

Testing: proceso llevado adelante por probadores de *software* (también conocidos como *testers*, su denominación en inglés) que planifican y llevan a cabo pruebas de *software* para comprobar si funcionan correctamente. Identifican el riesgo de sufrir errores de un *software*, los detectan y los comunican.

Wireframe: es un boceto donde se representa visualmente, de una forma muy sencilla y esquemática, la estructura del sistema. En los *wireframes* no se utilizan colores, tipografías ni elementos gráficos específicos.

WPF (Windows Presentation Foundation): es una tecnología de Microsoft presentada como parte de Windows Vista. Permite el desarrollo de interfaces de interacción en Windows tomando características de aplicaciones Windows y de aplicaciones web.

Anexo 3

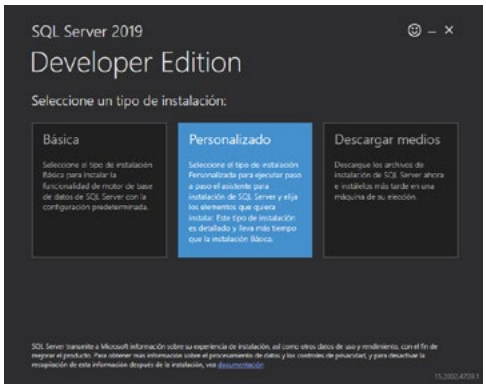
Instructivo de instalación de SQL Server 2019 - Versión Desarrollador

SQL Service Developer es una edición gratuita con todas las características que se puede usar como base de datos de desarrollo y pruebas en un entorno que no sea de producción.

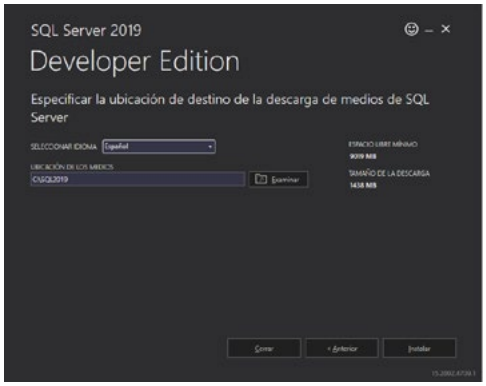
1. Descargar y comenzar la instalación de SQL Server 2019 Desarrollador desde este enlace de descarga.

Archivos

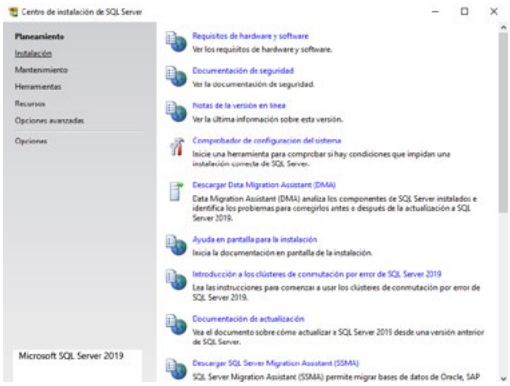
2. En el proceso de instalación, elegir la opción “Personalizado”.



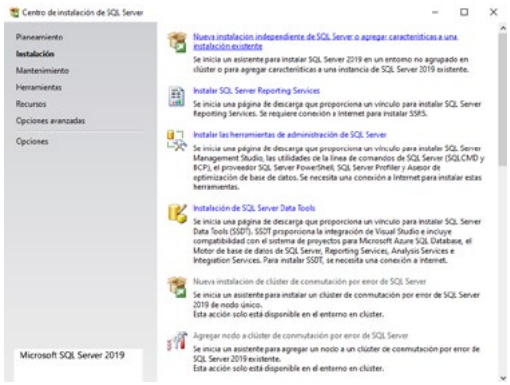
3. Elegir ubicación de instalación.



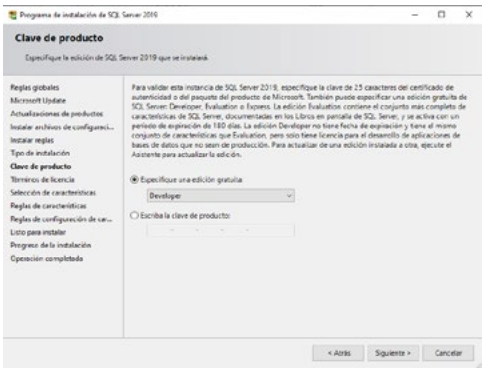
4. Hacer clic en “Instalación”.



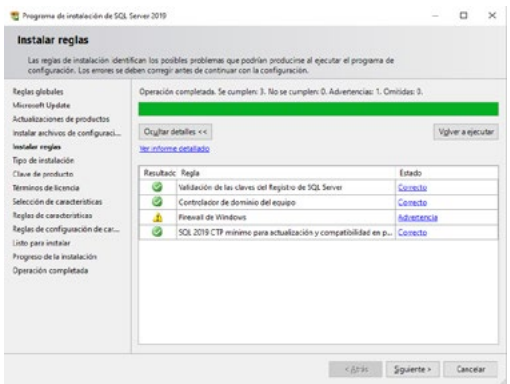
5. Seleccionar la primera opción.



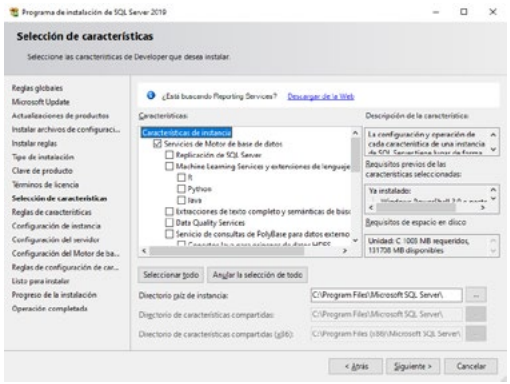
6. Seleccionar la opción “Especifique una edición gratuita” y luego “Developer”.



7. Verificar que los puntos figuren como correctos (más allá del *firewall*).

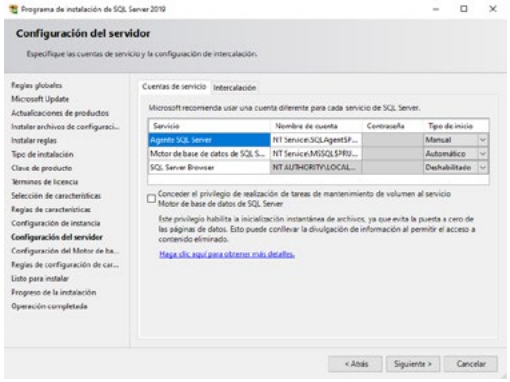


8. Activar opción “Servicios de Motor de base de datos”.



9. Configurar la instancia de la base de datos. Se puede definir un nombre o dejar el que viene por defecto.

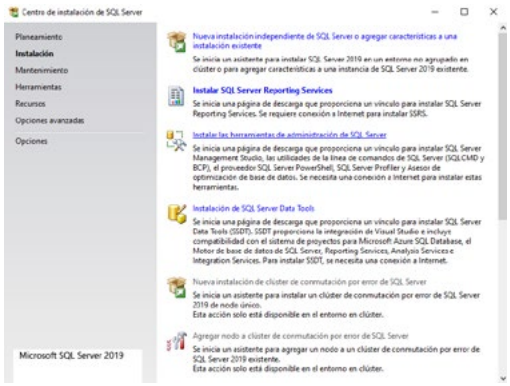
10. Ver la configuración. Pulsar “siguiente”.



11. Definir autenticación. Con autenticación de Windows o modo mixto, para el segundo caso deberán definir la clave (y no olvidarla).

12. Por último, presionar “instalar”.

13. Una vez que finalicen la instalación de SQL Server, deberán instalar las herramientas de administración de SQL Server.



Lo pueden instalar desde la opción seleccionada o desde los siguientes enlaces:

- Descarga de SQL Server Management Studio (SSMS).
- SSMS-Setup-ENU-exe.

Archivos

Imágenes

Créditos audiovisuales: bit.ly/38ySuco


BA
Buenos
Aires
Ciudad