

GUÍA PRÁCTICA

1. Datos Generales

|  |  |
| --- | --- |
| Carrera: | Tecnología Superior en Desarrollo de Software |
| Período académico: | Abril 2023 - Agosto 2023 |
| Asignatura: | Programación Visual |
| Unidad Nº: | 2 Patrón de diseño Modelo, Vista y Controlador. |
| Tema: | MVC |
| Ciclo-Paralelo: | M3A |
| Fecha de inicio de la Unidad: | 11/04/2023 |
| Fecha de fin de la Unidad | 10/08/2022 |
| Práctica Nº: | 2 |
| Horas: | 15 |
| Docente: | Ing. Patricio Pacheco |
| Nombre: | Jose David Muñoz Garcia |

1. Contenido

2.1 Introducción

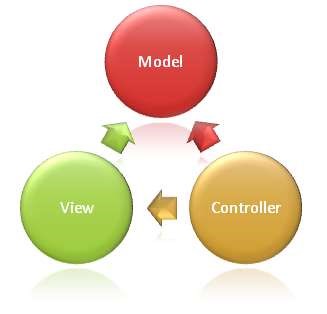
Modelo-vista-controlador (MVC) es un patrón de arquitectura de software, que separa los datos y la lógica de negocio de una aplicación de su representación y el módulo encargado de gestionar los eventos y las comunicaciones.

Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario.

Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

2.2

Objetivo de la Guía



Crear un proyecto utilizando Java y su componente gráfico Swing, aplicando el patrón de desarrollo Modelo – Vista – Controlador.

En cada componente colocar la clase correspondiente, de tal manera que no se coloque la funcionalidad equivocada

2.3 Materiales, herramientas, equipos y software

Equipos de computación con el IDE de netbeans,

SDK Java 8. (Swing)

Material Guía (Módulo, ejercicios prácticos).

2.4 Procedimiento

# En Netbeans, crear un nuevo proyecto de Java Application

2.

Creamos tres paquetes en el árbol principal del proyecto

llamados: modelo, vista y controlador.

En cada uno de estos paquetes se colocarán los elementos que

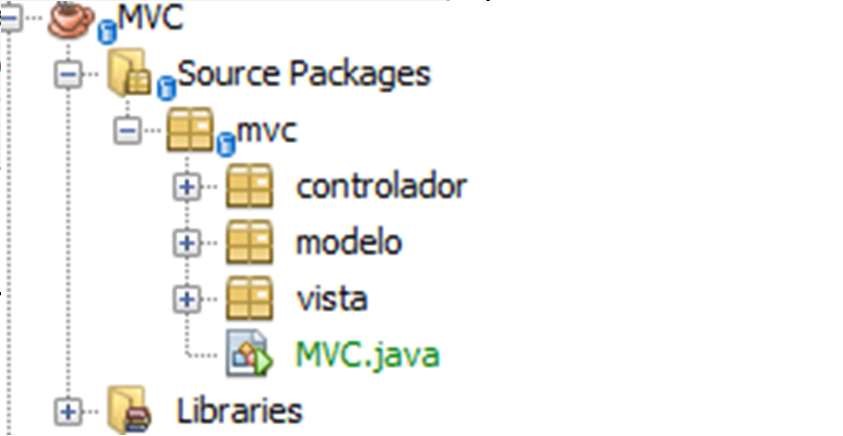
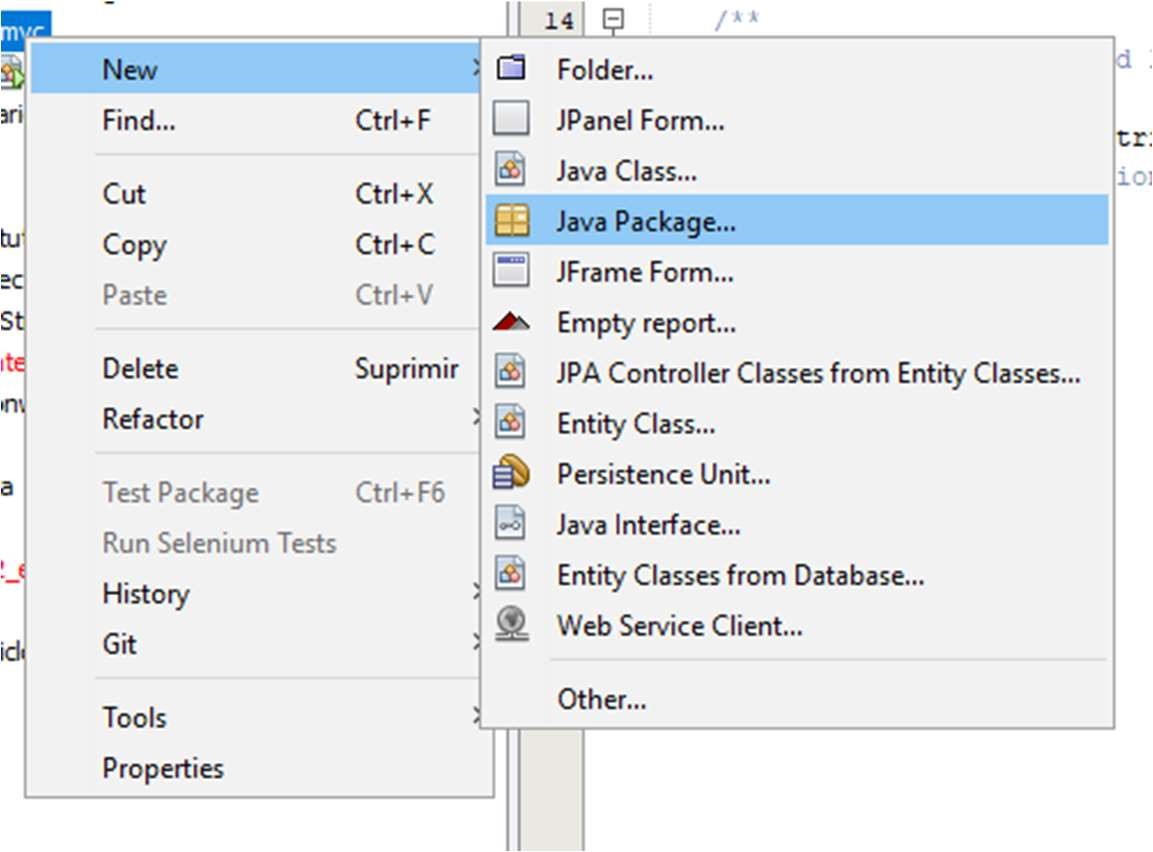
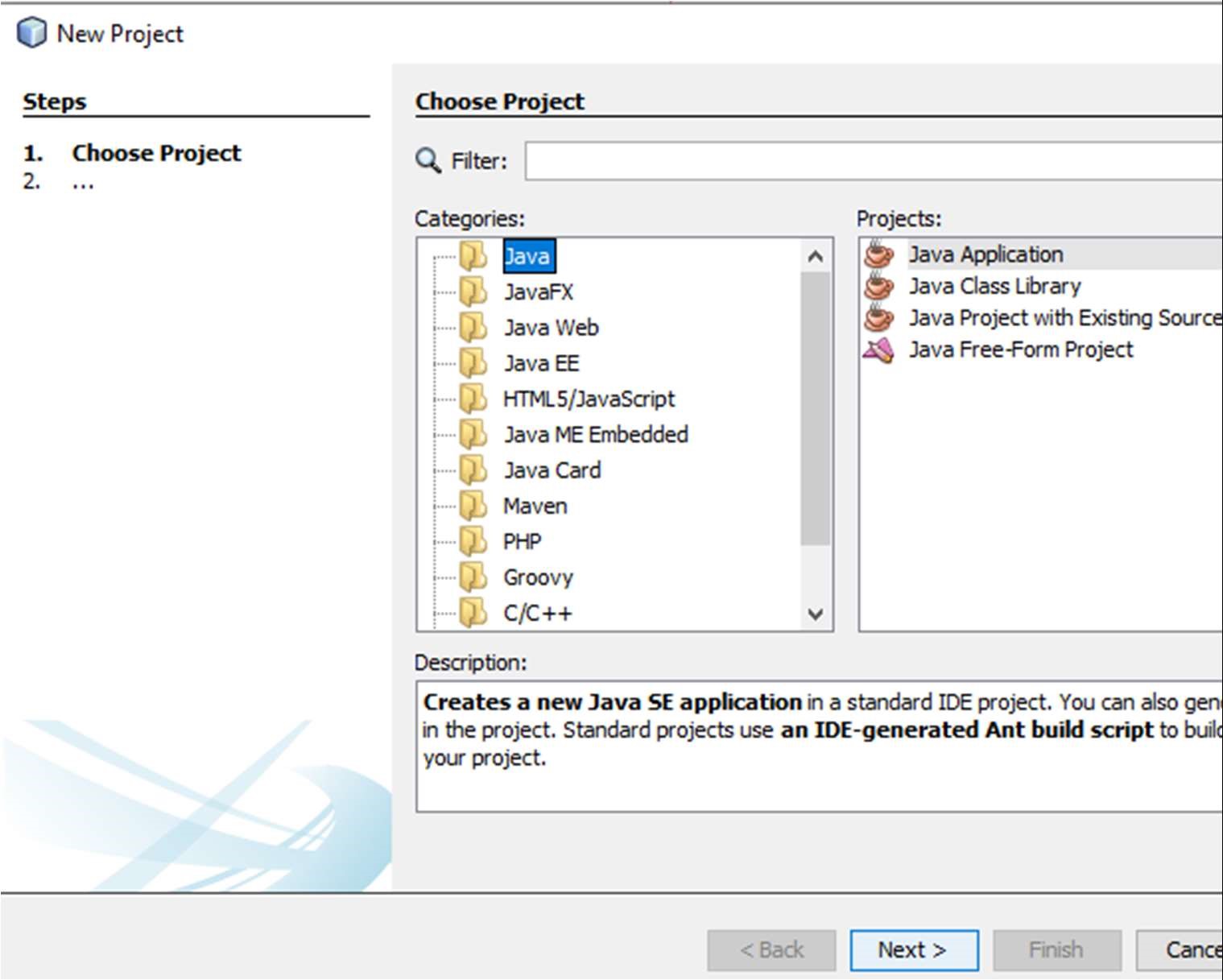
correspondan.

NOTA: No se podrá colocar en

ningún caso

código que no sea de

cada elemento.



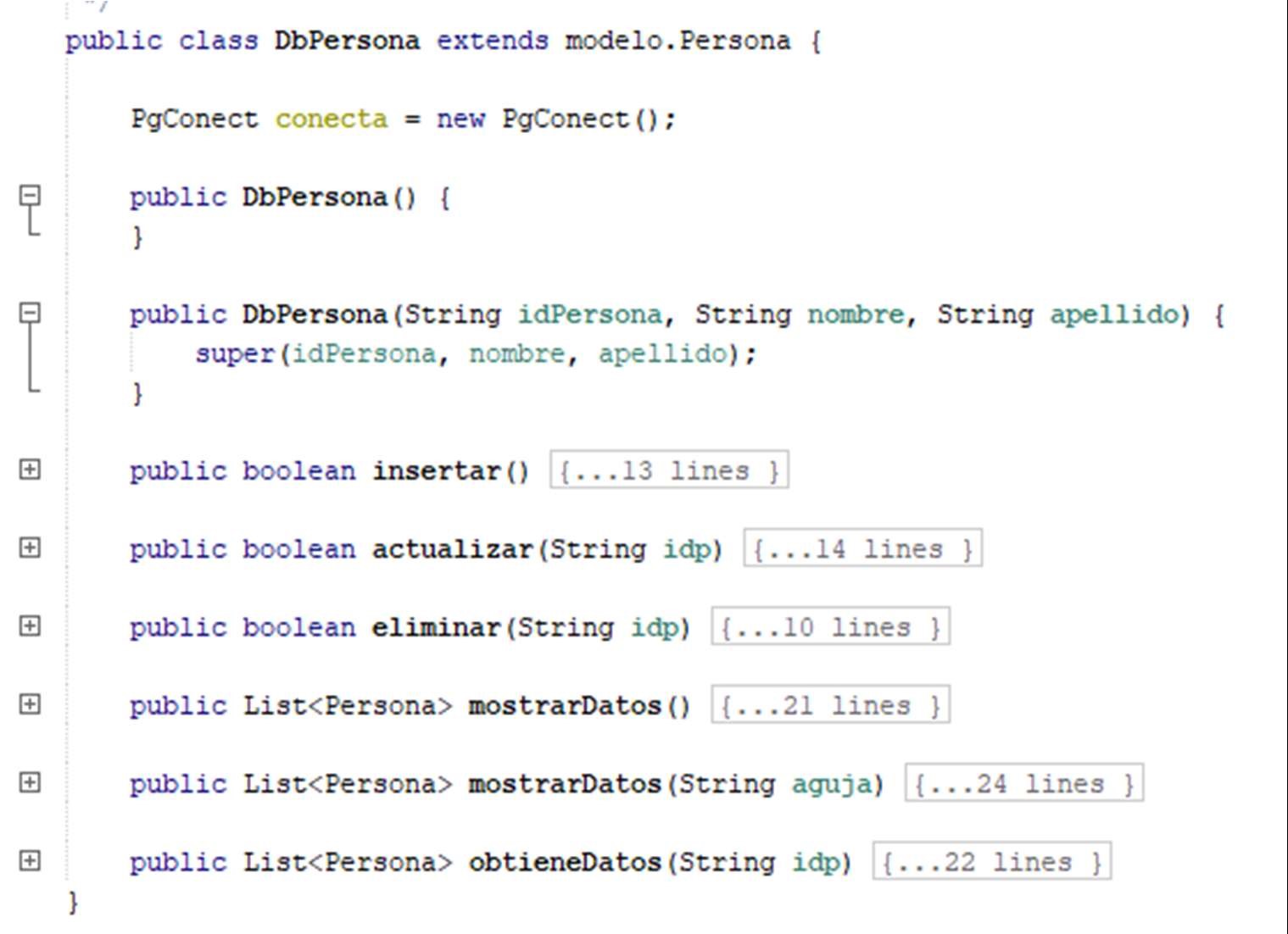
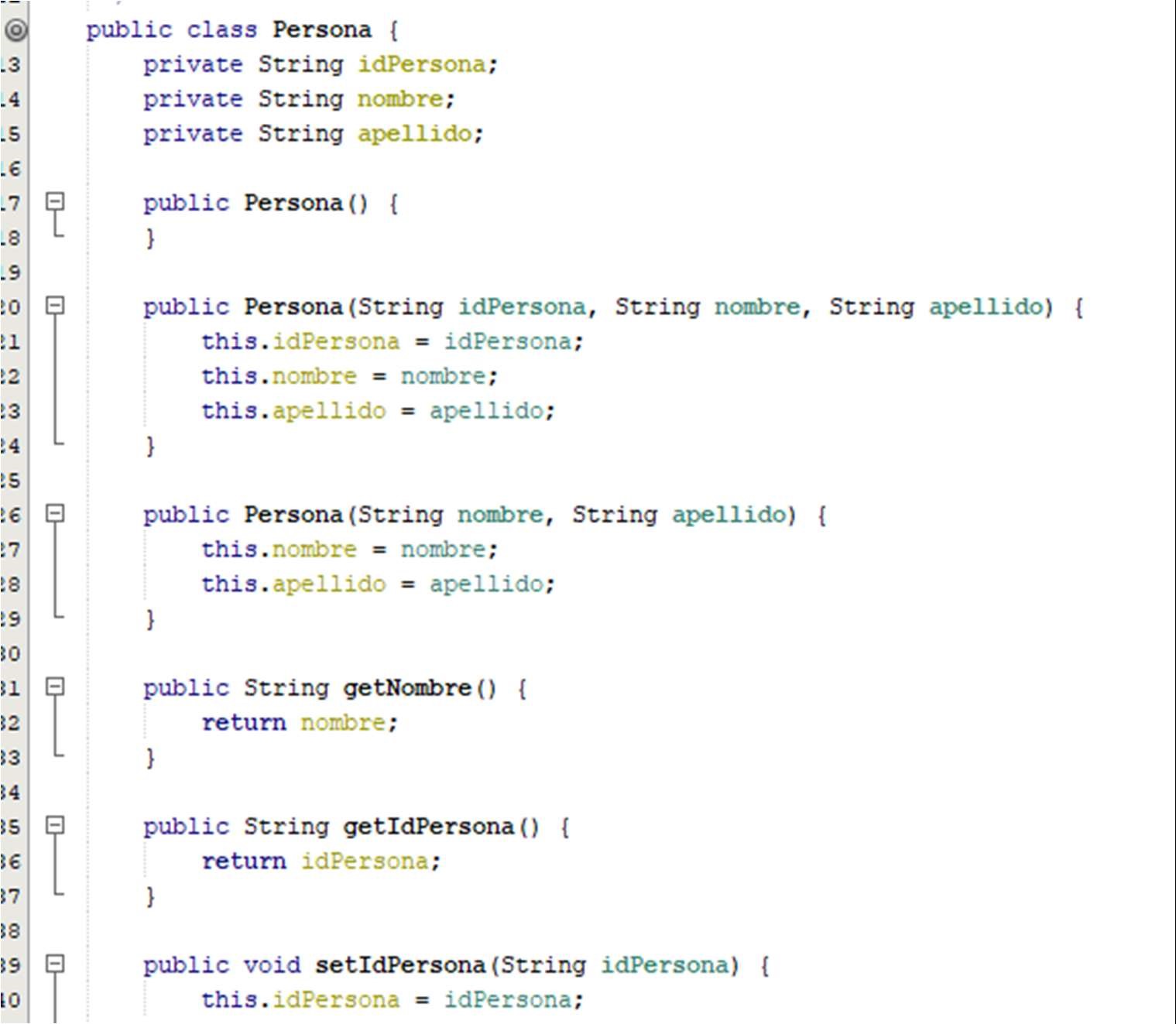
|  |  |
| --- | --- |
| 3. Dentro del paquete vista creamos la siguiente interfaz, con la paleta de Swing: | |
| 4. |  |
| Generar los getters y setters correspondientes a cada elemento que se va ha utilizar luego en el controlador.  NOTA: En este elemento no pude ir nada más que no sea el constructor y los getters y setters. |

5.

En el paquete modelo, creamos lo necesario para manejar los datos y la conexión

a la base de datos, en este ejemplo, creamos la clase Persona, DbPersona y

PgConnect



|  |  |  |
| --- | --- | --- |
|  | | |
|  |  | |
|  | |
| 6. Finalmente en el controlador, enlazamos los otros elementos y establecemos la lógica del negocio | | |
|  | |  |
|  |
|  | | |

2.5 Resultados esperados

|  |
| --- |
| Descripción en norma APA |
| Urma, R. G., Fusco, M., & Mycroft, A. (2014). Java 8 in Action: Lambdas, Streams, and functional-style programming. Manning Publications Co. |

Los estudiantes entienden y dominan el patrón de desarrollo Modelo Vista Controlador y son capaces de crear un proyecto de desarrollo Java Application desde cero, aplicando el patrón y colocando cada clase en su lugar apropiado sin salirse del esquema.

2.6 Bibliografía

Schildt, H. (2014). Java: the complete reference. McGraw-Hill Education Group.

Patrón de arquitectura Modelo Vista Controlador (MVC). (s. f.). Recuperado 10 de diciembre de 2019, de

https://web.archive.org/web/20130420133140/http://www.lab.inf.uc3m.es/~a0080802/RAI/mv c.html

**Desarrollo de la Guia practica**

**Pasos para implementar el patrón MVC**

A continuación, se detallan los pasos a seguir para implementar el patrón MVC en un proyecto de desarrollo:

Identificar y definir los elementos del modelo: Determine qué datos y lógica de negocio deben estar presentes en el modelo de su aplicación. Defina las clases y métodos necesarios para manejar estos aspectos.

Crear las clases de la vista: Diseñe la interfaz gráfica de usuario utilizando los componentes gráficos adecuados. Implemente las clases y métodos necesarios para mostrar los datos y capturar las interacciones del usuario.

Implementar las clases del controlador: Defina las clases y métodos que se encargarán de recibir las interacciones del usuario desde la vista, procesarlas y actualizar el modelo en consecuencia. También se encargarán de actualizar la vista con los cambios realizados en el modelo.

Establecer las relaciones entre los componentes: Conecte los elementos del modelo, la vista y el controlador, de modo que puedan comunicarse entre sí. Por ejemplo, el controlador debe tener acceso al modelo y a la vista para poder realizar las operaciones necesarias.

Probar y depurar el sistema: Realice pruebas exhaustivas para asegurarse de que el patrón MVC se haya implementado correctamente. Verifique que la comunicación entre los componentes funcione correctamente y que la aplicación se comporte según lo esperado.

**Beneficios del patrón MVC**

El patrón Modelo-Vista-Controlador ofrece varios beneficios para el desarrollo de aplicaciones:

Separación de responsabilidades: Permite separar claramente la lógica de negocio y los datos del manejo de la interfaz de usuario. Esto facilita el mantenimiento y la evolución de la aplicación, ya que cada componente puede modificarse de forma independiente.

Reutilización de código: Al separar la lógica de negocio y la interfaz de usuario, es posible reutilizar componentes en diferentes contextos. Por ejemplo, se puede utilizar el mismo modelo de datos en diferentes vistas sin necesidad de modificarlo.

Facilidad de mantenimiento: Debido a la división clara de responsabilidades, el mantenimiento de la aplicación se vuelve más sencillo. Es más fácil identificar y corregir errores, así como realizar mejoras y actualizaciones en cada componente por separado.

Flexibilidad y escalabilidad: El patrón MVC permite que la aplicación sea flexible y escalable. Es posible agregar nuevas funcionalidades o modificar las existentes sin afectar otros componentes. Esto facilita la adaptación de la aplicación a cambios en los requisitos o en el entorno de desarrollo.

Resultados esperados Al aplicar el patrón MVC en el desarrollo de una aplicación Java utilizando Swing, se espera que se logren comprender y dominar los conceptos y principios de este patrón de arquitectura de software. También se espera que puedan crear un proyecto desde cero, ubicando cada clase en su lugar correspondiente según el patrón MVC y sin violar su estructura. Esto permitirá desarrollar aplicaciones de manera eficiente, modular y fácilmente mantenible.

3. Firmas de Responsabilidad

|  |  |  |
| --- | --- | --- |
| ESTUDIANTE | DOCENTE | COORDINADOR DE CARRERA |
| Nombre: David Muñoz                Firma | Nombre: Ing. Patricio Pacheco                Firma | Nombre: Ing. Jessica Herrera                Firma |
| Fecha: | Fecha: 05/01/2022 | Fecha: |