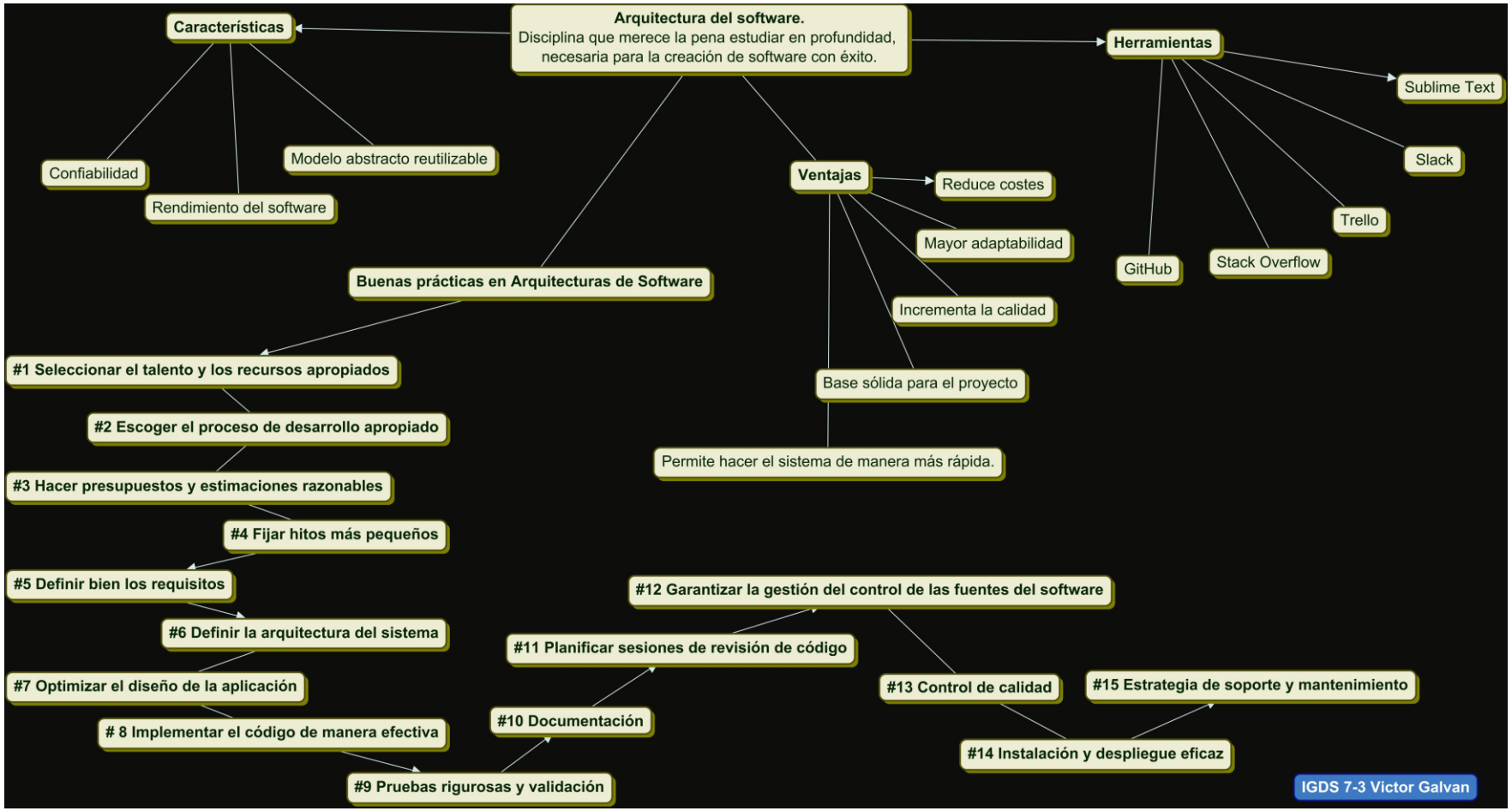
Guía para el examen

Arquitecturas de software (Concepto, ventajas y desventajas)



Tipos de arquitecturas (Conceptos de cada tipo y ejemplos)

Cliente-Servidor

Modelo de diseño de software en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes.

Ventajas

Escalabilidad
Fácil Mantenimiento

Desventajas

Bajo rendimiento
Costos elevados

Microservicios

Construir una aplicación como un conjunto de pequeños servicios, los cuales se ejecutan en su propio proceso y se comunican con mecanismos ligeros.

Ventajas

Facil de implementar
Reusabilidad del código
Idependencia de la aplicacion

Desventajas

Memoria
Difícil gestionar
Difícil de hacer tests

N capas

Objetivo primordial es la separación de las partes que componen un sistema software o también una arquitectura cliente-servidor.

Ventajas

Portable
Curva de aprendizaje reducida

Desventajas

Dificultar para diseñar las capas
Menor eficiencia

Dirigida a eventos

Patrón de arquitectura software que promueve la producción, detección, consumo de, y reacción a eventos. Un evento puede ser definido como "un cambio significativo en un estado".

Ventajas

Muy escalable
Sin dependencia entre servicios

Desventajas

El servidor no estará al tanto si un servicio se cayo
Sin comprobación ante la atención de eventos

Microkernel

También conocido como arquitectura de Plug-in, permite crear aplicaciones extensibles, mediante la cual es posible agregar nueva funcionalidad mediante la adición de pequeños plugins que extienden la funcionalidad inicial del sistema.

Ventajas

Buen despliegue
Buena testibilidad
Mayor rendimiento

Desventajas

No son fáciles de desarrollar
El Core debe estipular muy claramente cómo se desarrollarán los plugings

Serverless

Manera de crear y ejecutar aplicaciones y servicios sin tener que administrar infraestructura. Su aplicación continúa ejecutándose en servidores, pero terceros se encargan de toda la administración de los servidores.

Ventajas

Costos menores
Atención a lo importante
Menor administración de recursos

Desventajas

Complejas
Tiempos de carga y latencia
Muy difícil de controlar correctamente

Requerimientos funcionales y no funcionales (Concepto de ambos y ejemplos)

FUNCIONALES

Un requisito funcional define una función del sistema de software o sus componentes.

NO FUNCIONALES

Especifican criterios para evaluar la operación de un servicio de tecnología de información, en contraste con los requerimientos funcionales que especifican los comportamientos específicos.

	A	B	C	D	E	F
1	Nombre del proyecto: MYKEY ASOCIADOS					
2	Encargado: HUGO MOROYOQUI			Fecha: 12-ENERO-2022		
3	Codigo	Descripción				Modulo
4	RF-1	INICIO DE SESIÓN POR MEDIO DE CORREO Y CONTRASEÑA				AUTENTICACIÓN
5	RF-2	FUNCIÓN PARA RECORDAR CREDENCIALES DE ACCESO				AUTENTICACIÓN
6	RF-3	VALIDAR QUE USUARIO SEA TIPO TÉCNICO				AUTENTICACIÓN
7	RF-4	VALIDAR QUE EL USUARIO NO ESTE DESHABILITADO				AUTENTICACIÓN
8	RF-5	OPCIÓN PARA REDIRECCIONAR AL REGISTRO				AUTENTICACIÓN
9	RNF-1	APLICAR MASCARA AL CAMPO CONTRASEÑA PARA QUE NO SEA VISIBLE EL VALOR				AUTENTICACIÓN
10	RNF-2	UTILIZAR MATERIAL DESIGN EN LA INTERFAZ				GENERAL
11	RNF-3	LA APLICACIÓN DEBE CONTENER LOS COLORES DE LA EMPRESA				GENERAL
12	RNF-4	LAS CONTRASEÑAS DE USUARIO DEBEN ESTAR ENCRIPTADAS				AUTENTICACIÓN
13	RNF-5	LA CONTRASEÑA UNICAMENTE SE PUEDE CAMBIAR SOLICITANDO DIRECTAMENTE AL AREA DE DESARROLLO DE SOFTWARE DE LA EMPRESA				AUTENTICACIÓN