Trabajo Práctico: Objetivos

Ejercicios teóricos:

1. Explica qué es npm y su función en un proyecto de Node.js.

Npm (Node Package Manager) es el sistema de gestión de paquetes de Node.js. Su principal función es ayudar a los desarrolladores a gestionar las dependencias (librerías, herramientas, módulos, etc.) que un proyecto de Node.js necesita para funcionar correctamente. Gestión de dependencias, Automatización de tareas, Distribución de paquetes, Manejo de versiones, Integración con el ecosistema de Node.js

2. Describe la estructura básica del archivo package.json y menciona tres campos importantes.

```
{
 "name": "nombre-del-proyecto",
 "version": "1.0.0",
 "description": "Descripción del proyecto",
 "main": "index.js",  Archivo principal del proyecto
 "scripts": {
  "start": "node index.js",  Script para iniciar la aplicación
  "test": "echo \"Error: no test specified\" && exit 1"   Script de prueba
 },
 "dependencies": { Dependencias del proyecto
  "express": "^4.17.1"
},
 "devDependencies": { Dependencias de desarrollo
  "nodemon": "^2.0.7"
 },
 "keywords": ["node", "express", "npm"], Palabras clave que describen tu proyecto
 "author": "Nombre del autor",
 "license": "ISC" Especifica la licencia bajo la cual se distribuye el código de tu proyecto
}
```

- Keywords: Este campo es un array de palabras clave que describen tu proyecto o paquete. Son útiles para ayudar a otros desarrolladores a encontrar tu proyecto cuando buscan términos relacionados en el registro de npm o en buscadores. Las palabras clave pueden estar relacionadas con el propósito del proyecto, las tecnologías utilizadas o los casos de uso.

- License: Este campo especifica la licencia bajo la cual se distribuye el código de tu proyecto. Indica los derechos y restricciones de uso para otros usuarios que quieran utilizar, modificar o distribuir tu código. Usar una licencia es fundamental para dejar claro cómo los demás pueden interactuar legalmente con tu proyecto.

- Dependencies: Este campo contiene un objeto con las dependencias que el proyecto necesita para su funcionamiento en producción. Cada clave representa el nombre de una dependencia y su valor es la versión especificada. Estas dependencias se instalan con el comando npm install.

3. ¿Cuál es la diferencia entre dependencias normales y dependencias de desarrollo en npm?

Las dependencias normales son las librerías o paquetes que son necesarios para que el proyecto funcione en producción. Estas son las dependencias que tu aplicación necesita para ejecutar sus funcionalidades en un entorno real, ya sea en un servidor, en el navegador o en cualquier otro entorno donde se ejecute tu código. Ej: npm install

Las dependencias de desarrollo son los paquetes que son necesarios sólo durante el desarrollo y la construcción del proyecto, pero no son necesarias para la ejecución del proyecto en producción. Estos paquetes suelen ser herramientas como compiladores, herramientas de testing, linters, etc. Ej: npm install --save-dev

4. ¿Qué comando utilizamos para actualizar un paquete instalado en nuestro proyecto?

Si deseas actualizar un paquete específico a su última versión compatible según el package.json, puedes usar el siguiente comando: *npm update <nombre-del-paquete>*

5. ¿Para que se utiliza el comando npm link?

El comando npm link se utiliza para crear un enlace simbólico (o "link") entre un paquete npm local y un proyecto que lo utiliza, permitiendo probar y desarrollar ese paquete localmente sin tener que publicarlo en el registro de npm.

6. Ver el módulo de inicio de un proyecto