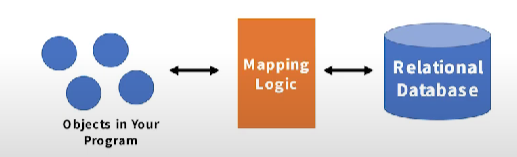
1. **Sequelize ORM**

**Картина, която съдържа текст, часовник

Описанието е генерирано автоматично**

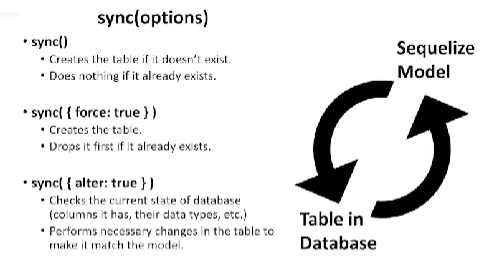
Sequelize e модул, който позволява на Javascript разработчиците да работят с релационна база от данни по-лесен начин. Например, вместо да пишем голям брой редове sql код за да създадем някаква таблица, със Sequelize това е възможно да стане по-прост и разбираем начин. Sequelize e съвместим с доста системи за управление на база от данни. Например: MySQL, SQL Sever, SQLite, PostgresSQL и MariaDB. Специфичното определение на Sequelize e: ORM (Object relational mapper) или преведено буквално на български: Обектно релационен картографер. С други думи ORM е техника, която позволява да се манипулират данните в база от данни по обектно ориентиран начин. Можем да представим таблица в база от данни като обект в Sequelize и като използваме методите на този обект, можем да извършваме самата манипулация



Sequelize може да се добави към Node.JS проект, чрез package manager. Най-използваният е NPM, затова ще въведем командата npm install sequelize за да добавим зависимостта (dependency) към package.json файла,който на кратко съдържа важни метаданни за нашия проект(име,автор,скриптове,версия и т.н.) и нашите зависимости.

За да използваме Sequelize с MySQL трябва да инсталираме също зависимостта mysql2. Това е драйвъра, който позволява да правим заявки, манипулации към MySQL база данни от Node.JS програма.

* **Синхронизация на модел**



1. **Vue.JS**

****

Това е Javascript Framework и също екосистема, която покрива основните характеристики, които са нужни за разработването на потребителски интерфейси.

* *Framework – Колекция от Javascript библиотеки, които предоставят на разработчик „шаблон“ ,който обслужва често срещаните задачи при програмиране на JavaScript апликация. С други думи framework(софтуерна рамка) е структура, която предоставя конкретен контектс и ти помага да създадеш приложение в рамките на този контекст. Софтуерните рамки предоставят правила и насоки. Чрез тези правила, разработчик може да създаде комплексно приложение много по-бързо и експедитивно, отколкото да го построим от нулата. Софтуерната рамка ни позволява да се фокусираме върху същинските проблеми за построяването на нашата апликация като напр. бизнес логиката.*

* *Декларативно и императивно програмиране – Чрез декларативно програмиране описваме свойствата на желания ни резултат, а чрез императивно се описва по какъв начин и чрез какви команди да се постигне дадения резултат.*

Vue предоставя декларитивен и компонентно-базиран програмен модел, който позволява да се постигне максимална продуктивност и минимални разходи. Едни от основните черти на Vue са:

* Declarative Rendering (Декларативно изобразяване) – Vue разширява стандартния HTML със синтаксиса template, който позволява декларативно да опишем нашия HTML изход въз основа на състоянията на Javascript кода.
* Reactivity (Реактивност) – Vue автоматично проследява промените в състоянието на JavaScript и ефективно акуализира DOM, когато настъпят промени.

Vue също осигурява разширени функционалности, необходими за изграждането на комплексно приложение като routing, state management и build tooling

* *Routing – механизъм, който насочва HTTP заявки към кода, който ги обработва. С прости думи в Router, ти определяш какво ще се случи, когато потребител посети дадена уеб страница или въведе определен URL адрес.*
* *State management(Управление на състоянието) – e имплемнтация на шаблон за дизайн(Design Pattern), който ни позволява да синхронизираме нашите състояния(states) с всички компоненти на приложението, което създаваме. Състояние може да се определи като нещо, което е част от нашата апликация. Като например списък с потребители, списък с продукти или всякакви данни, които идват от някаква база от данни.*
* *Build tooling – Vue също поддържа инструмент, който автоматизира създаването на Vue проект като изтегля всички необходими файлове, които са нужни за правилната работоспособност на VUE и ги структурира по специфичен начин. Известни Build tools за VUE са webpack и Vite. Характерно също за тези инструменти е че те идват и с сървър за development. Основната употреба на тези сървъри е, когато искаме да виждаме резултатите на нашия проект, когато работим по него локално, т.е. използва се когато приложението ни е в процес на разработка.*

Vue най-често се прилага, когато е решено да се създаде Single Page Application (SPA), но не са изключени и възможнoстите да се създадат приложения със Server-Side-Rendering(SSR), Static-Site-Generation (SSG), мобилни апликации, desktop, WebGL или терминални.

* *Single Page Application – Това е уеб приложение, което взаимодейства с потребителя чрез динамично пренаписване на DOM-а(текущата уеб страница) с нови данни от уеб сървъра. Като е характерно, че при първото посещаване на определения уеб сайт се извлича скелет на html и в самия html файл са описани CSS и Javascript файловете, и според действията на потребителя, страницата се пренаписва чрез Javascript логиката, без да се правят нови заявки към уеб сървъра за нов ресурс(страница).*

Структурата на \*.vue файл е подобна на HTML файлов формат, наречен Single-File Component(SFC). VUE SFC, както подсказва името, енкапсулира логиката на компонента (JavaScript), шаблона(html), и стилизирането(CSS) в един файл.

SFC формат:

Картина, която съдържа текст

Описанието е генерирано автоматично

1. **Node.JS**



Node.js е кросплатформена среда за изпълнение с отворен код за разработване на сървърни и мрежови приложения. Приложенията Node.js са написани на JavaScript и могат да се изпълняват в рамките на изпълнение на Node.js на OS X, Microsoft Windows и Linux.

Node.js също така предоставя богата библиотека от различни JavaScript модули, което опростява разработването на уеб приложения, използващи Node.js до голяма степен. Може също да се каже, че Node.JS е с най-голямата екосистема от библиотеки сравнение с конкуренцията.

Подобни среди за разработване на сървърни и мрежови приложения са: ASP.NET, Django, Ruby on Rails и други.

Приложенията с Node.js работят в един процес, без да създава нова нишка за всяка заявка. Node.js предоставя набор от асинхронни I/O примитиви в стандартната си библиотека, които предотвратяват блокирането на JavaScript код и като цяло библиотеките в Node.js са написани с помощта на неблокиращи парадигми, което прави блокиращото поведение по-скоро изключение, отколкото норма.

Node.js има уникално предимство, защото фронтенд разработчиците, които пишат JavaScript за браузъра, вече могат да пишат кода от страна на сървъра в допълнение към кода от страна на клиента, без да е необходимо да научават напълно различен език.

Javascript кода в Node.js runtime средата се изпълнява чрез Google v8 engine.

1. **Express.JS**



Express.JS е framework изграден на Node.JS, който улеснява работата ни с Node. Express e предназначен за изграждане на уеб приложения и API-та и е определен, като най-популярния server-side framework. Създаването на Back-end от нулата за по-комплексно приложение в Node.js може да се окаже доста досадно и скъпо за разработване, тъй като сами трябва да напишем целия стандартен код за обработка на различни HTTP заявки, настройки на портове, route handlers и т.н. Чрез използване на уеб framework като Express.js, разработчиците могат да спестят време и да се съсредоточат върху други важни задачи, които касаят самия проект. Express поддържа routing и middlewares

* *Routing - Маршрутизирането се отнася до определяне как дадено приложение отговаря на клиентска заявка според определената крайна точка, която е URI (или път) и също метода за HTTP заявка.*
* *Middleware -Middlewares са функции, които се изпълняват “по-средата“ на цикъла заявка-отговр между клиент и сървър, с цел екзекуция на някакво действие. Те имат достъп до обекта заявка (req), обекта на отговор (res) и следващата middleware функция в цикъла заявка-отговор на приложението. Следващата middleware функция обикновено се обозначава с променлива, наречена next.*

*Функциите на междинния софтуер могат да изпълняват следните задачи:*

* + *Изпълнете произволен код преди заявката или отговора да е получен, с цел някаква манипулация.*
  + *Направата промени в обектите на заявката и отговора.*
  + *Прекратяване цикъла заявка-отговор.*
  + *Извикване на следващата middleware функция в стека.*

Докато самият Express е доста минималистичен, общността и разработчици са създали съвместими middleware пакети, за да се справят с почти всеки проблем при уеб разработка. Има библиотеки за работа с бисквитки, сесии, потребителски влизания, URL параметри, POST данни, header-и за сигурност и много други.

1. **NPM**

**Картина, която съдържа текст, графична колекция

Описанието е генерирано автоматично**

npm (първоначално съкращение от Node Package Manager) е мениджър на пакети за езика за програмиране JavaScript. npm е мениджърът на пакети по подразбиране за средата на изпълнение на JavaScript Node.js. Състои се от command line клиент, наричан още npm, и онлайн база данни с публични и платени частни пакети, наречена регистър на npm. Достъпът до регистъра се осъществява чрез клиента, а наличните пакети могат да бъдат преглеждани и търсени чрез уебсайта на npm.

Друга важна употреба на npm е за контрол на версиите и също менажиране на нашите зависимисти(dependencies).

Чрез въведената команда в терминала “npm init“, можем да създадем файл към проекта, който се нарича package.json. Този файл съдържа важни метаданни, които описват нашия проект като версия, име на автор, различни скриптове, нашите зависимости и други. NPM може да се сравни с Maven в Java средата. И двата инструмента имат много прилики помежду си.

1. **MySQL**



MySQL е система за управление на релационни бази данни с отворен код. Той се основава на езика за структурни заявки (SQL), който се използва за добавяне, премахване и модифициране на информация в базата данни. Стандартните SQL команди, като ADD, DROP, INSERT и UPDATE могат да се използват с MySQL.

MySQL може да се използва за различни приложения, но най-често намира употреба при уеб сървърите. Уебсайт, който използва MySQL, може да включва уеб страници, които имат достъп до информация от база данни. Тези страници често се наричат „динамични“, което означава, че съдържанието на всяка страница се генерира от база данни при зареждането на страницата. Уебсайтовете, които използват динамични уеб страници, често се наричат уебсайтове, управлявани от база данни.

Много уебсайтове, управлявани от база данни, които използват MySQL, също използват уеб скриптов език като PHP или JavaScript за достъп до информация от базата данни. MySQL командите могат да бъдат включени в PHP или Javascript кода, позволявайки част или цялата уеб страница да бъде генерирана от информация от базата данни.