**Desktop app tools onderzoeks document**

Inhoud

[Introductie 3](#_Toc177114843)

[Literatuur onderzoek 4](#_Toc177114844)

[Electron.js 4](#_Toc177114845)

[Tauri 4](#_Toc177114846)

[WPF of WinForms 5](#_Toc177114847)

[Java met JavaFX 5](#_Toc177114848)

[Qt 5](#_Toc177114849)

[methodologie 6](#_Toc177114850)

[Resultaat 7](#_Toc177114851)

[Conclusie 8](#_Toc177114852)

[Referenties 9](#_Toc177114853)

# Introductie

Mijn naam is Martijn verhappen. Ik studeer hbo ICT bij fontys hogeschool en zit in semester drie. Voor dit semester is het de bedoeling dat ik een individuele opdracht maak die twee front-end applicaties heeft. Hier heb ik de keuze gemaakt om een desktop applicatie te maken voor de manager kant van een web shop. Dit document zal informatie bevatten over welke frameworks het belangrijkste zijn voor de desktop applicatie voor mijn opdracht en welke van de frameworks ik ga gebruiken.

# Literatuur onderzoek

Allereerste moet ik uitzoeken welke tools er beschikbaar zijn voor het maken van een desktop applicatie en wat de voordelen en nadelen zijn bij hun gebruik.

Om ervoor te zorgen dat het literatuur onderzoek niet te lang wordt ga ik 5 populaire desktop applicatie development tools bekeiken. Deze tools zijn: Electron.js, Tauri, C# met WPF of winforms, Java met JavaFX en Qt (C++/Python).

Ter verduidelijking geen van de frameworks eisen een back-end gecodeerd in een specifiek codeer taal. Wanneer er bij een framework staat dat het kennis van een specifieke taal vereist dan is dit voor het gebruik van de framework zelf en niet dat het alleen kan communiceren met het gespecificeerde codeer taal.

## Electron.js

Electron.js meestal gewoon Electron genoemd is een framework voor het maken van desktop applicaties met gebruik van javascript, HTML en CSS. Door het embedden van node.js en Chromium is er maar één codebase nodig voor het creëren van een multi-platform desktop applicatie en vereist geen ervaring met native development (*Introduction | Electron*, n.d.).

Electron is een veelgebruikt framework ook door grootte bedrijven zoals discord, wordpress en whatsapp.

Electron is wel een zware framework om een desktop applicatie mee te maken omdat het node.js en Chromium gebruikt om multi-platform te zijn.

## Tauri

Tauri is een toolkit voor het maken van desktop applicaties dat gebouwt is op rust en node.js. Tauri kan met bijna elke front-end programeer taal samenwerken waaronder HTML, CSS, Javascript, next.js, sveltekit en meer (*What Is Tauri? | Tauri Apps*, n.d.).

Omdat het gemaakt is door gebruik van Rust is dit ook nodig voor het gebruikt van Tauri en WebView2 is nodig voor het genereren van web content op windows en moet daarom geinstaleert zijn om met Tauri een desktop app te maken.

Tauri is veel lichter dan Electron maar is relatief nieuw dus zal niet zo veel features hebben als Electron.

## WPF of WinForms

WPF (Windows Presentation Foundation) en WinForms zijn frameworks die gemaakt zijn voor windows alleen. Dit betekent dat een desktop applicatie gemaakt is met deze frameworks ze alleen functioneel zulle zijn op windows machines. Er zijn wel verschillen tussen WPF en WinForms. Het grootste verschil is dat WinForms gebruik maakt van de basis windows controls terwijl WPF dit niet doet maar, dit verschil valt alleen te merken als je met beide frameworks hebt gewerkt (*WPF Vs. WinForms - the Complete WPF Tutorial*, n.d.).

Het verschil tussen WPF en WinForms is hierdoor de flexibiliteit van het framework waarin WPF de flexibelere is.

Het grootste nadeel van deze twee frameworks is dat het alleen werkt op windows en niet linux of MacOS.

## Java met JavaFX

JavaFX is een framework gebouwd op de programeer taal Java en werkt op windows, linux en MacOS. JavaFX was gemaakt om een moderne toolkit voor client applicaties beschikbaar te maken. Omdat er een grootte groep individuen en bedrijven samengewerkt hebben met het maken van dit framework is er veel documentatie te vinden over het gebruik van dit framework (*JavaFX*, n.d.).

Omdat het een taal gemaakt is op Java is kennis over Java nodig om goed gebruik te kunnen maken van dit framework. De UI styling is ook complexer dan de meeste web technologieën.

## Qt

Qt uitgesproken als het engelse woord cute is een desktop app development framework die werkt op c++ met de mogelijkheid om python bindings te gebruiken. Dit zorgt ervoor dat je cross-platform applicaties kan maken en veel controle hebt over de UI elementen van een desktop applicatie (*Desktop App Development Tools & UI Design | QT*, n.d.).

Je moet wel C++ en python gebruiken voor Qt en is te complex voor gebruik van een simpele desktop applicatie.

# methodologie

Om de vraag welke desktop applicatie framework kan ik het beste gebruiken bij het maken van de manager kant van een web shop heb ik naar vijf frameworks gekeken en waar zij voor dienen. Hiervoor heb ik de voor- en nadelen van elk framework onderzocht zodat ik aan kan geven welk framework het beste werkt voor mijn project daarna, zal ik een kleine prototype maken van het literatuur onderzoek uitgekozen framework en als deze bevalt gebruik ik hem in mijn eigen project.

# Resultaat

In dit hoofdstuk noem ik alle frameworks die ik niet ga gebruiken op en leg uit waarom ze niet gekozen zijn voor mijn project. In het hoofdstuk conclusie leg ik uit waarom ik het overgebleven framework zal gebruiken.

Electron.js: Electron is zwaarder dan de andere frameworks en biedt niet veel unieks aan voor mij om te zeggen dat het gebruik ervan waardevol genoeg is.

WFP of WinForms: Ik behouw het liefst de mogelijkheid voor mijn programma om cross-platform te zijn zodat ik hier later mischien nog iets mee kan doen.

Java met JavaFX: omdat ik een andere front-end ga maken in react wat ik nog moet leren lijkt het mij niet handig om een framework te gebruiken waarvan er gezegd wordt dat UI styling complex is.

Qt: Ik ken geen c++ en heb een heel basische kennis over python en samengevoegd met het moeten leren van react lijkt het mij niet verstandig om Qt te gebruiken.

# Conclusie

Voor het maken van mijn desktop applicatie ga ik gebruik maken van Tauri omdat het een lichtere versie van Electron en daarom perfect is voor het maken van de manager kant van mijn web shop. Deze applicatie wordt niet groot en hoeft niet uitertematen veel features te hebben dus het fijt dat Tauri die minder heeft dan Electron zal geen probleem moeten worden.

# Referenties

*Introduction | Electron*. (n.d.). <https://www.electronjs.org/docs/latest>

*What is Tauri? | Tauri Apps*. (n.d.). <https://tauri.app/about/intro/#:~:text=Tauri%20is%20a%20toolkit%20that,and%20the%20CLI%20leverages%20Node>.

*WPF vs. WinForms - The complete WPF tutorial*. (n.d.). <https://wpf-tutorial.com/nl/2/over-wpf/wpf-vs-winforms/>

*JavaFX*. (n.d.). <https://openjfx.io/>

*Desktop App development Tools & UI Design | QT*. (n.d.). <https://www.qt.io/product/qt-for-desktop>