The Importance of Bundler Features in React Development

Hello and thanks for taking the time to respond to this form! With your answers I'll be able to better narrow which bundlers would be a good fit for use within Codifly for my Bachelor's thesis. Below are the features that I have at the moment determined to have potential impact on the choice, and tend to differ between bundlers. I'll be asking you to assign a level of importance to each of these features. I have also provided a brief description and if sometimes a link to more information for each of these as well. If you are not sure what a feature means you can refer to its description.

If I left something out, or you have additional remarks, there will also be an extra text area at the end.
I'm always available in person or on Teams as well for questions or remarks ;)

Answers are anonymous.

Thanks again!

* Vereist

List of Features

Code-splitting:
With code-splitting, instead of one main bundle being created, code is split into chunks which then can be lazy-loaded by the browser using dynamic imports. When combined with HTTP/2s multiplexing, the performance impact of fetching multiple chunks at the same time is also significantly reduced. For more info, see https://reactjs.org/docs/code-splitting.html

Zero-configuration:
A bundler advertised as zero-configuration should be close to being plug and play. This does not have any influence on the extent of customisation, only on initial setup.

High speed bundling:
The performance of the actual bundling in development, so during eg. yarn run dev.

Customisability:
Certain bundlers are more opinionated in how they are configured than others. Webpack offers extensive customisation, for example. Zero-configuration bundlers tend to be less customisable as a whole. This does not mean that they lack features, but that the find details might not be as tuneable.

ES5 Support:
Most bundlers transpile down to ES5 with their built-in transpilers. A few might (ESBuild) need a little extra help to transpile down to ES5, and can only transpile to ES6 by default. Caniuse reports support of ES6 for 98.31% of Belgian users.

Dynamic expressions in import:
Eg. await import(`./${variable}.js`);
Some bundlers can get around this by bundling everything that could possibly be targeted by the statement together with the rest, but others cannot do this. As a result this example would not work with every bundler.

Typescript typechecking:
Some bundlers provide a plugin for typechecking TypeScript during builds. Others rely on you having the proper IDE setup for type-checking. Note that this in no way means they can't transpile Typescript, just that they don't typecheck it.

Hot module replacement:
HMR helps during development by only reloading modules that get changed, this means you don't need to reload, or have the dev server reload the window, and the state of your app can be retained.

1. How important are below features? *

	I don't know/ I'm not sure	Not important at all	Not really important	Has its uses	Very important	This is essential
Code-splitting	◯	◯	◯	◯	◯	◯
Zero-configuration	◯	◯	◯	◯	◯	◯
High speed bundling	◯	◯	◯	◯	◯	◯
Customisability	◯	◯	◯	◯	◯	◯
ES5 Support	◯	◯	◯	◯	◯	◯
Dynamic expressions in import	◯	◯	◯	◯	◯	◯
Typescript typechecking	◯	◯	◯	◯	◯	◯
Hot module replacement	◯	◯	◯	◯	◯	◯

2. Are there other features which you consider to be important, or just want to highlight? (Optional)

3. Do you have any other remarks or feedback? (Optional)