**Generate Package of Vagaro Toolkit**

**To create a React Package toolkit that bundles all components into a single package for use in Vagaro projects, follow these steps:**

1. Create the React Components

* Develop reusable React components based on Vagaro's standards.
* Organize components into a structured folder hierarchy, ensuring they are modular and reusable.

2. Setup Package Repository

* Initialize a new Git repository for your toolkit.
* Set up a package.json file using npm init, specifying metadata like name, version, and dependencies.
* Make sure all components are exported from an index.js file in the root folder so that users can import them easily.

3. Build a Package for Distribution

* Use tools like Rollup or Webpack to bundle the components into a single library.
* Configure the bundler to support ES6 and CommonJS formats for compatibility with different projects.
* Add scripts to package.json to simplify building (npm run build) and testing the package.

4. Write Documentation

* Create clear documentation explaining how to use each component.
* Include examples and describe any configuration or customization options.

5. Versioning and Publishing

* Use semantic versioning to manage updates and changes.
* Tag and push your commits to the Git repository.
* Optionally, use GitHub actions for CI/CD to automate build, test, and deployment processes.

6. Publishing to npm (Optional)

* If you want to make the toolkit public or available via npm, publish it using npm publish.
* Ensure that the project’s package.json includes proper fields like main, module, and types if you're using TypeScript.

7. Integrating in Vagaro Projects

* Add the package as a dependency in Vagaro projects using npm install or yarn add.

**Lets Do this Step by Step**

1. **Configure package.json**

In the package.json file, the @your\_organization name must match your GitHub organization or username, and the package\_name must match the GitHub repository name.

"name": "@your\_organization/package\_name",

"url": "git+https://github.com/your\_username/repo\_name.git"

Here for Vagaro Toolkit

{

"name": "@vagaroo/vagaro-react-toolkit",

"version": "1.0.0",

"main": "dist/index.js",

"types": "dist/index.d.ts",

"files": [

"dist",

"README.md"

],

"repository": {

"type": "git",

"url": "git+[https://github.com/vagaroo/vagaro-react-toolkit.git"](https://github.com/vagaroo/vagaro-react-toolkit.git%22)

},

"publishConfig": {

"registry": "<https://npm.pkg.github.com/>"

},

"peerDependencies": {

"react": "^17.0.0 || ^18.0.0",

"react-dom": "^17.0.0 || ^18.0.0"

},

"dependencies": {

},

"scripts": {

"start": "webpack serve --config webpack.config.js",

"build": "webpack --config webpack.config.js && tsc",

..........,

"build-storybook": "storybook build"

},

........,

"devDependencies": {

..........,

"@types/react": "^18.3.10",

"@types/react-dom": "^18.3.0",

"copy-webpack-plugin": "^12.0.2",

"css-loader": "^7.1.2",

"sass": "^1.79.4",

"sass-loader": "^16.0.2",

"ts-loader": "^9.5.1",

"typescript": "^5.6.2",

"webpack": "^5.95.0",

"webpack-cli": "^5.1.4"

}

}

1. **List out all the Component in Index.js file which you want to make Package**

Here is Example of 3 Component which needs to be make as Package Library.

export { VgTextbox } from './stories/components/VgTextbox/VgTextbox';

export { VgCheckbox } from './stories/components/VgCheckbox/VgCheckbox';

export { VgButton } from './stories/components/VgButton/VgButton';

3. **Install Webpack and related dependencies:**

Here is all 3 Below dependencies must be add.

npm install --save-dev typescript webpack webpack-cli ts-loader @types/react @types/react-dom --force

npm install sass sass-loader@latest css-loader@latest --save-dev --force

npm install copy-webpack-plugin --save-dev -- force

4. **Create a Webpack configuration file (webpack.config.js) in root directory**

Add Content in **webpack.config.js file**

const path = require('path');

const CopyPlugin = require('copy-webpack-plugin');

module.exports = {

entry: './src/index.ts',

mode: 'development', // or 'production' based on your need

output: {

filename: 'index.js',

path: path.resolve(\_\_dirname, 'dist'),

libraryTarget: 'umd',

library: '@vagaroo/vagaro-react-toolkit',

umdNamedDefine: true

},

resolve: {

extensions: ['.ts', '.tsx', '.js', '.jsx',]

},

module: {

rules: [

{

test: /\.tsx?$/,

use: 'ts-loader',

exclude: /node\_modules/

},

{

test: /\.jsx?$/,

exclude: /node\_modules/,

use: {

loader: 'babel-loader',

options: {

presets: ['@babel/preset-react', '@babel/preset-env'],

},

},

},

{

test: /\.s[ac]ss$/i,

use: [

'style-loader', // Injects styles into DOM

'css-loader', // Turns CSS into JS modules

'sass-loader', // Compiles Sass to CSS

],

},

{

test: /\.css$/,

use: ['style-loader', 'css-loader'],

},

{

test: /\.(png|jpg|gif|svg)$/,

use: [

{

loader: 'file-loader',

options: {

name: '[name].[ext]',

outputPath: 'assets/images',

},

},

],

},

]

},

externals: {

react: 'react',

'react-dom': 'react-dom'

},

plugins: [

new CopyPlugin({

patterns: [

{ from: 'src/logo.svg', to: 'logo.svg' }, // Copy logo.svg if needed

{ from: 'src/App.css', to: 'App.css' },

{ from: 'src/index.css', to: 'index.css' },

{ from: 'src/components', to: 'components' },

{ from: 'src/common', to: 'common', },

{ from: 'src/utils', to: 'utils' },

],

}),

],

};

5. **Create a Personal Access Token (PAT) on GitHub:**

* Setup your Github Account
  + Navigate to [GitHub's website](https://github.com).
  + Click on **Sign up** and provide a username, email address, and password.
  + Follow the prompts to complete the registration process, which includes verifying your email address.
  + After account creation, you can configure your profile, set up two-factor authentication for added security, and explore various GitHub features.
* After Logged in, Go to **GitHub Settings > Developer settings > Personal access tokens**
* Click "**Generate new token**"
* Select the **write:packages** scope
* Copy the generated token E.g (ghp\_0Ik6nkShIhzSxEoPWSvCQ2O6Ks4qh50AvM43ar)

6**. Authenticate with GitHub Packages: Create or edit “.npmrc” file in your project root:**

Create .npmrc file and Add below content in **.npmrc** file (if already Exist then replace)

//npm.pkg.github.com/:\_authToken=YOUR\_PERSONAL\_ACCESS\_TOKEN

@ your\_username:registry=https://npm.pkg.github.com

7. **Build your project:**

npm run build

* This will Transfer your source files from src to dist. It will create dist folder.
* Now, Add this dist folder path in **.gitignore** file. Like this:

# production

/build

/dist

8. **Verify Output:**

* After running the build command, check the **dist** directory to ensure that **index.js** and any other necessary files are exist.
* Ensure **main** in **package.json** correctly points to your entry file in the **dist** directory. Here is example

"main": "dist/index.js",

This configuration helps identify the end user by providing the main entry point of the library when it is installed."**6. Publish Your Package to npm**

**9. Versioning:** Update package.json file with proper version number. After every change in your package update the version number and build package and publish in npm. Follow this link for versioning. [About semantic versioning | npm Docs (npmjs.com)](https://docs.npmjs.com/about-semantic-versioning)

Here is Example of Package.json

"version": "1.0.1",

6.1 Login to npm:

If you haven't logged in to npm from the command line, do so: (First signup in npm then login using this link. [npm | Sign In (npmjs.com).](https://www.npmjs.com/login))

npm login

6.2 Publish your package:

To publish your package to npm and link it with the GitHub repository:

- Using this command you can publish your package.

npm publish

**7. Push your code to Github**

To publish your package to npm using a GitHub repository as the source, you'll first need to ensure that your package is built correctly, and then you can publish it to npm, specifying the GitHub repository.

- Add **.gitignore** file, and **“README.md”** in your root directory. In **.gitignore** file, you can add node\_modules, .env, etc. Files that you don't want push in your git repo.

7.1. Create a GitHub Repository

1. Go to GitHub and log in. ([GitHub: Let’s build from here · GitHub](https://github.com/))
2. Click the "+" button in the top-right corner and select **New repository**.
3. Name your repository and set it to **private** as needed.
4. Click **Create repository**.

7.2. Initialize Git in Your React App

- If you haven't initialized Git in your React app yet, navigate to your React project folder and initialize a Git repository:

cd your-react-app

git init

7.3. Add Files to Git:

Add all the files in your project to Git's staging area:

git add .

7.4. Commit Changes:

Commit your changes to the repository:

git commit -m "Initial commit"

7.5. Add GitHub Remote and Push to Repository:

Link your local project to the GitHub repository you just created and push your code:

git remote add origin https://github.com/your-username/repo\_name.git

git branch -M main

git push -u origin main

**-** if any error comes after **git push -u origin main** this command then

1. Open the .git/config file in a text editor. In VS Code, you can open it by running:  
 **code .git/config**

2. Update **.git/config** file like this and save file.

[core]repositoryformatversion = 0

filemode = true

bare = false

logallrefupdates = true

[remote "origin"]

url = https://your\_username@github.com/your\_username/repo\_name.git

fetch = +refs/heads/*\*:refs/remotes/origin/\**

[branch "main"]

remote = origin

merge = refs/heads/main

3. **Authenticate with the Personal Access Token** **(Token):**

* When you push changes, Git will prompt you for a username and password.
* Enter your GitHub username as the username.
* For the password, use the PAT **(Token)** you generated.

**8. Using Your Package in Another Project**

8.1. Authentication with Github Packages:

**-** Create or edit a **.npmrc** file in another project’s root and add the content below in **.npmrc** file.

//npm.pkg.github.com/:\_authToken=YOUR\_PERSONAL\_ACCESS\_TOKEN

@your\_username:registry=https://npm.pkg.github.com

**OR**

- In replace of create **.npmrc file**, you can set these two commands in your project’s root directory terminal:

npm config set //npm.pkg.github.com/:\_authToken=${YOUR\_PERSONAL\_ACCESS\_TOKEN}

npm config set @your\_username:registry=https://npm.pkg.github.com

-- **but authentication using .npmrc file is recommended approach.**

8.2. Package Installation:

- Using this command, you can install this package in another project.

npm install @your\_organization/package\_name

Here package name is **"@your\_organization/package\_name"** from **name** in **package.json** file.

8.3. Import Components from Package:

Import components from your package and use them in another project. Like this:

import { VgButton, VgCheckbox, VgTextbox } from '@your\_organization/package\_name';

function App() {

return (

<div className="App">

<div>

<VgTextbox

focusBorder

label="First Name"

placeholder="First Name"

required

/>

<VgButton

icon="faPlus"

iconPosition="prefix"

onClick={() => {}}

text="Submit"

type="blue"

/>

<VgCheckbox

checked

label="Check me"

onChange={() => {}}

/>

</div>

</div>

);

}

export default App;