Basic Frame

Create React App

[create-react-app](https://github.com/facebook/create-react-app) is a official CLI tool to create React apps. It’s based on jest , eslint, webpack and React, developed by facebook. It can provide a basic environment for developer to fast build React apps.

React

[React](https://reactjs.org/) is a JavaScript library for building user interfaces. It provide JSX syntax to write DOM structure in JavaScript. It’s very convenient and straight.

Typescript

[Typescript](https://www.typescriptlang.org/) is a programing language developed by Microsoft. It's brings static type checking to JavaScript code.It’s superset of JavaScript. It can reduce communication cost between different team or projects.

CRACO

[Craco](https://github.com/gsoft-inc/craco) is a tool to customized create-react-app. We can optimize antd load and add headers to webpack dev server by using this tool.

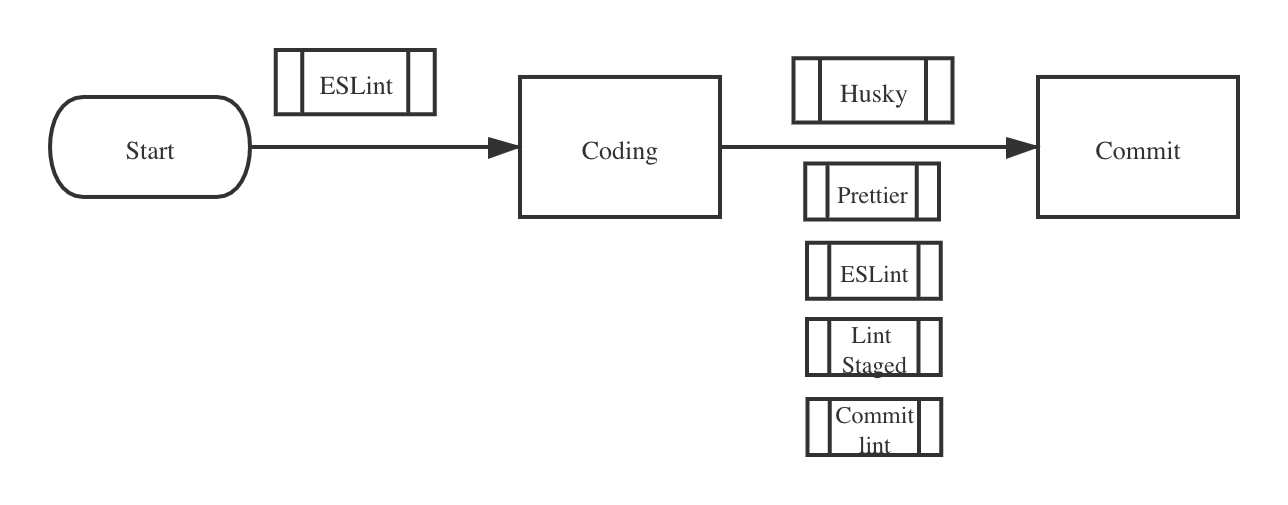
Jest

[Jest](https://jestjs.io/) is a JavaScript Testing Framework developed by facebook. It provide assert and mock by default. For now we only add some functional test to protect core module safety.

Styled-components

[Styled-components](https://styled-components.com/) is a CSS-In-JS solution for creating styles in javascript. Styles are create in runtime. We don’t need to care about the class name conflict.

Coding



ESLint

[Eslint](https://eslint.org/) is a static code analytics tool. It provide by create-react-app. It can find potential errors based our rules. We use **eslint-config-react-app** and eslint-config-prettier as our main rules. We can receive the error message if we write something wrong.

Prettier

[Prettier](https://prettier.io/) is a code format tool. We can format code according the special rule. For now we use the default rule. Normally it means 2 spaces for indent, always need semicolons and double quotes.

Stylelint

[Stylelint](https://stylelint.io/) is a static code analytics tool. It can combine with styled-components. Validate style code in same conventions.

Lint-staged

Lint-staged is a validation tool to check if git staging area code validate. For js and tsx files, we add prettier format and eslint check to keep commit files conform rules.

Commitlint

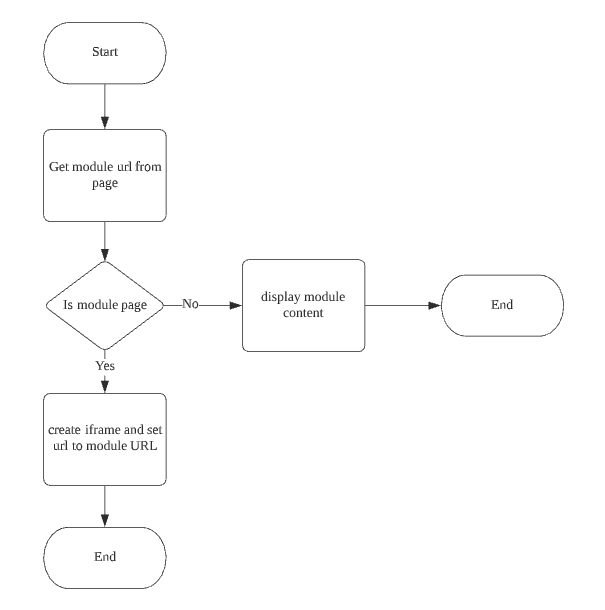
[Commitlint](https://github.com/conventional-changelog/commitlint) is a commit message lint tool in order to provide a unified commit message format.

Husky

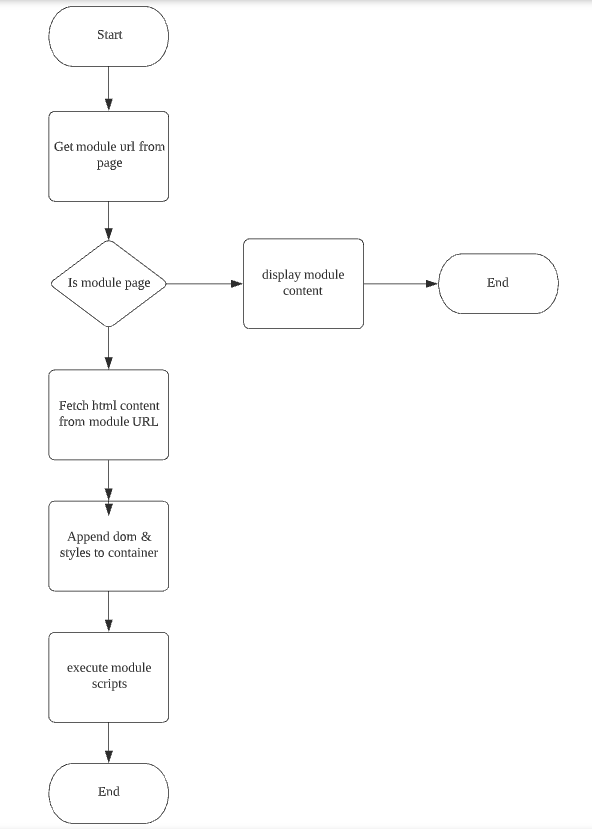
[Husky](https://github.com/typicode/husky) is a tool to prevent bad commits, push. It’s based on Git hooks. We place all rules to .huskyrc.js. We add two rules to this file. The first one is run lint-staged before commit. The second one is check commit message valid when commit.

Page modularization

* IFrame



* [qiankun](https://qiankun.umijs.org/) (based on [single-spa](https://single-spa.js.org/))



Props and Cons

|  |  |  |
| --- | --- | --- |
|  | iFrame | qiankun |
| Third part API | No limit | Need add CORS head |
| URL navigation | Need connection with container | Native support |
| Style isolate | Native support | Will affect container |
| Fullscreen modal | Hard to customize | Native Support |
| Page Scroll | Inner iFrame | Fullpage |
| Security | Native Support | Hart to keep safe |
| [HMR](https://webpack.js.org/concepts/hot-module-replacement/) | Native Support | No |
| Extra CORS header | No | Need allow container origin |
| Restrict API call | [CSP](https://developer.mozilla.org/zh-CN/docs/Web/HTTP/Headers/Content-Security-Policy/connect-src) | No |

Conclusion

For iFrame:

1. Mini-app run in a isolated runtime. It’s hard to display information outside self.
2. Mini-app run in a iframe container, we don’t know the total height. It’s hard to use browser scrollbar.
3. We can add [sandbox](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/iframe) property to restrict Mini-app access browser native API.
4. Add global style in Mini-app won’t effect the outside style. It will reduce the style bug If Mini-app include inappropriate code.
5. We don’t need to add extra CORS header to allow browser load this app.
6. iFrame is suitable for third-part developer. We don’t need them export special files. They only need upload their app with index.html.

For qiankun:

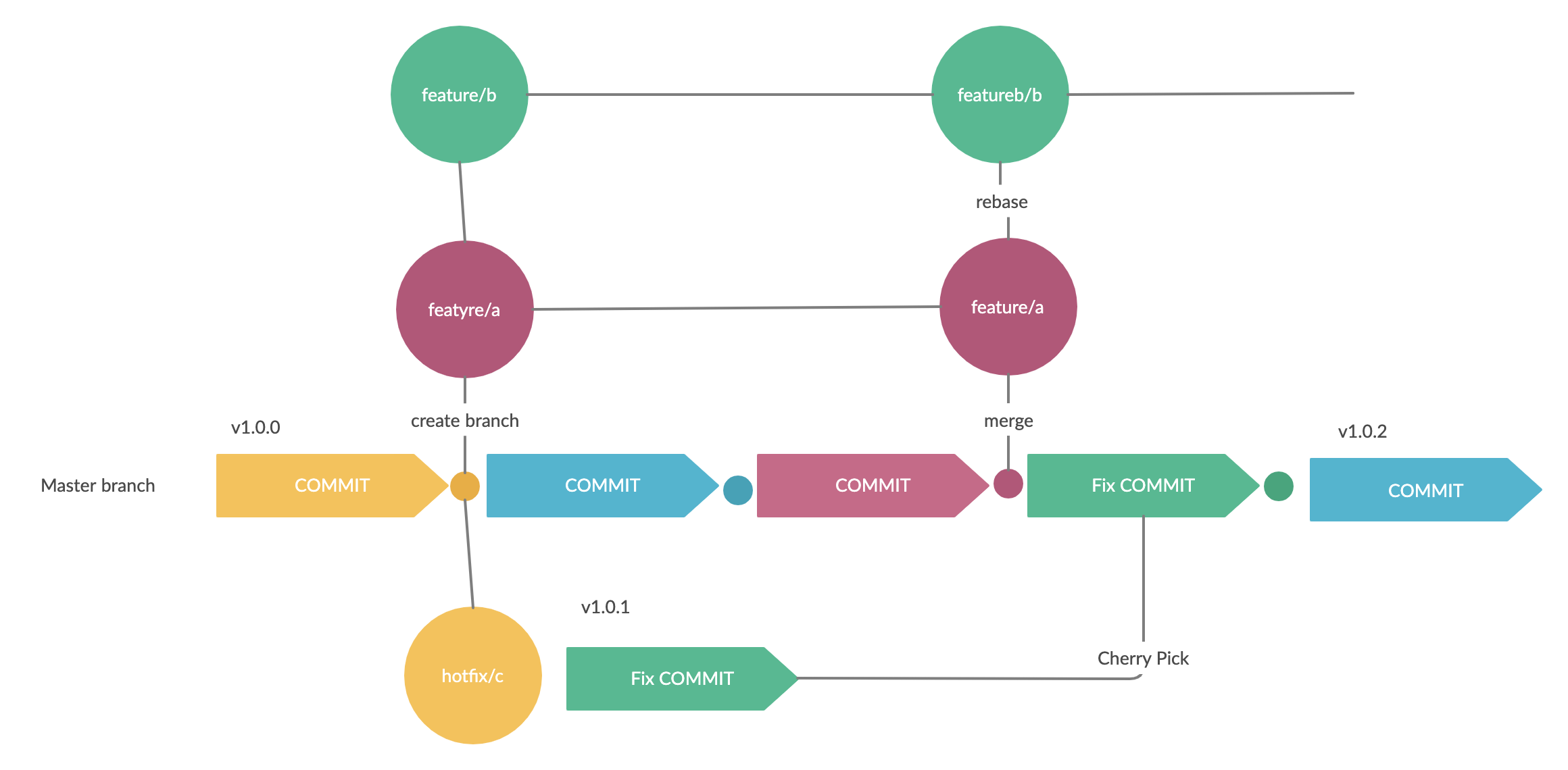
1. Mini-app run with main frame. We make a fake sandbox to isolate global variables. It’s easy to display information in main frame. The main frame will be break if the mini-app throw some error.
2. Mini-app use the same DOM tree with main frame. We don’t care about the height of app content. We can use the navtive scrollbar.
3. Mini-app share the same javascript sandbox with main frame. It’s hard to restrict app access browser native API.
4. Mini-a have a pp better experience just like develop by native React. App can deploy isolatedly, don’t need to update main frame. It’s more suitable for building a big app across multi internal team.

Scenarios:

1. For internal tools (no need multiple squad collaboration) – react directly
2. For office platform apps (may need multiple squad collaboration) - consider use qiankun or react
3. For platform apps (need third-part developers collaboration) - prefer use iFrame

Workflow

Github Flow

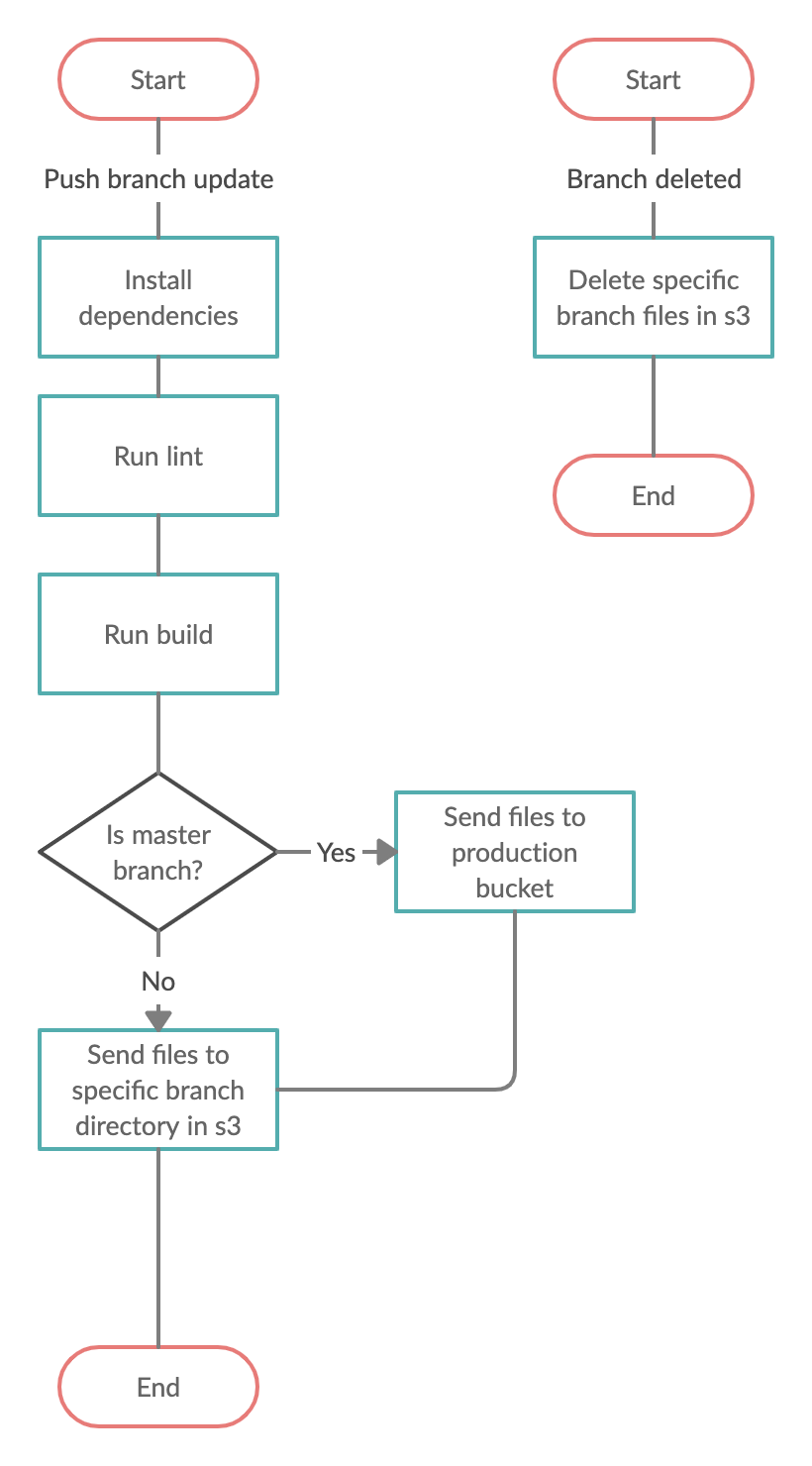


[Github flow](https://guides.github.com/introduction/flow/) is easy to understand. We use master branch to build production code. The master branch is always ready to online. We will create environments by branch name. So we always treat master branch as staging environment, and other branch as development environment.

Deployment

Gitlab CI/CD + AWS(Lamda + CloudFront + S3)

Gitlab CI/CD workflow



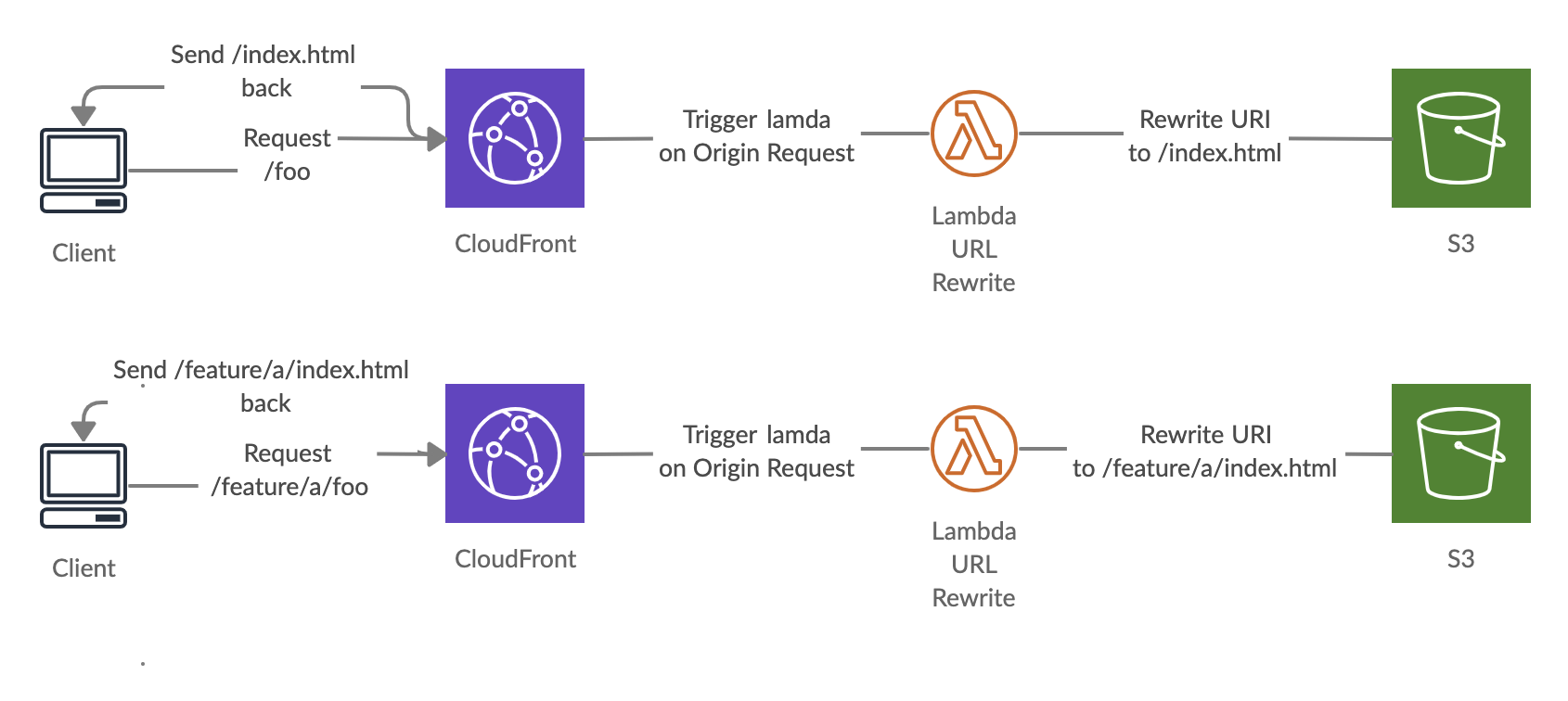
For branch update:

1. Will install the dependencies and cache them use md5 of yarn.lock content as cache key
2. Will run lint and may run unit test (later)
3. Will build source code to runtime code.
4. Will send build results to S3 dev bucket
5. Will send build results to S3 production bucket if we need to deploy to online.

For branch delete:

1. Will delete specific branch files in S3 dev bucket

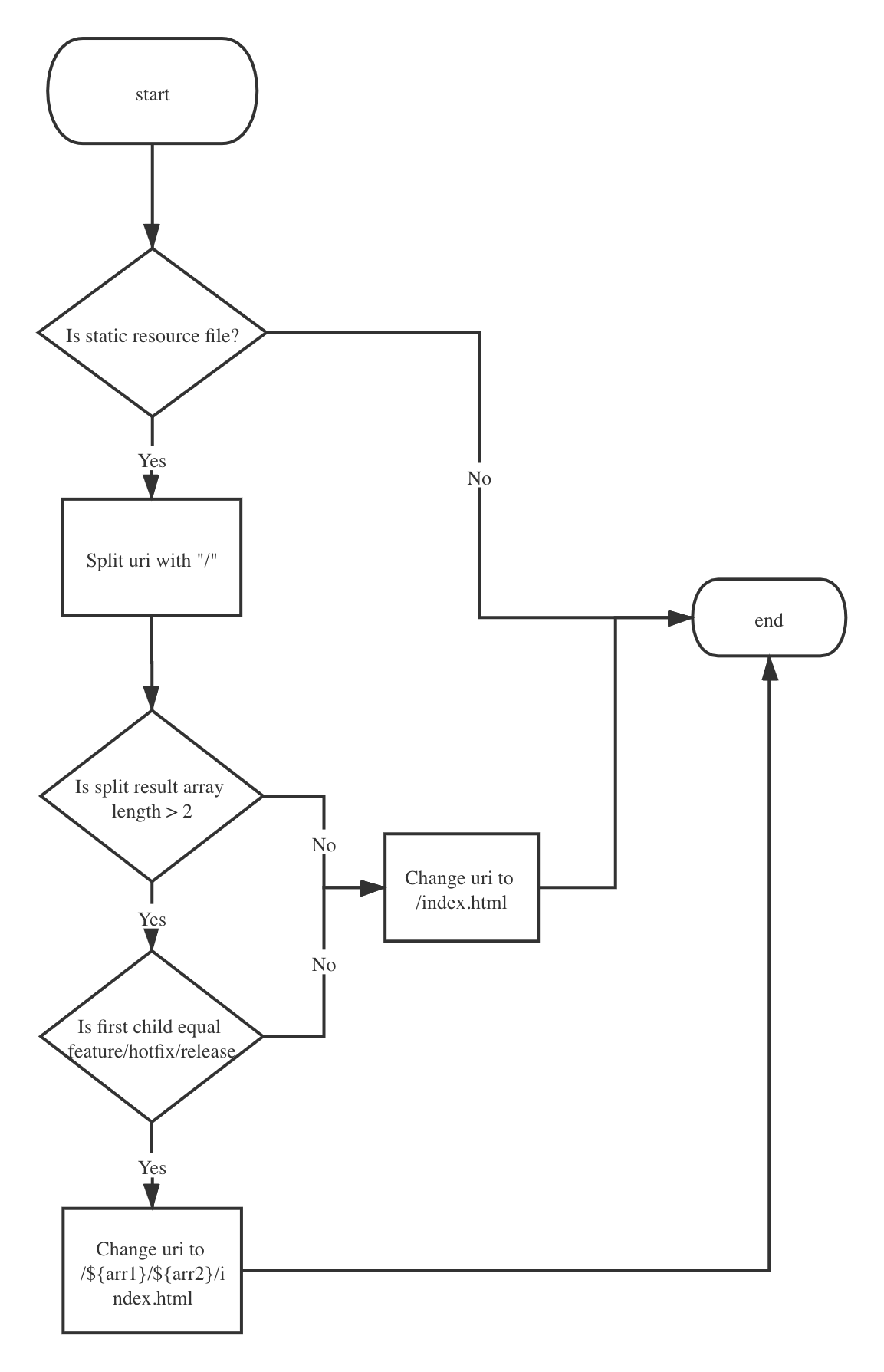
CloudFront + Lambda + S3 web hosting workflow



**CloudFront**

Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency, high transfer speeds, all within a developer-friendly environment. CloudFront will delivery S3 objects to Edge Server. So no matter S3 region select, it will use the fastest one.

**Lambda**



Lambda is service-less service. It can combine with CloudFront to modify request or response. We use it to rewrite request uri. So that all path request will send to index.html, use SPA for routing.

For static resources request (like .js,.css): we won’t modify the URI.

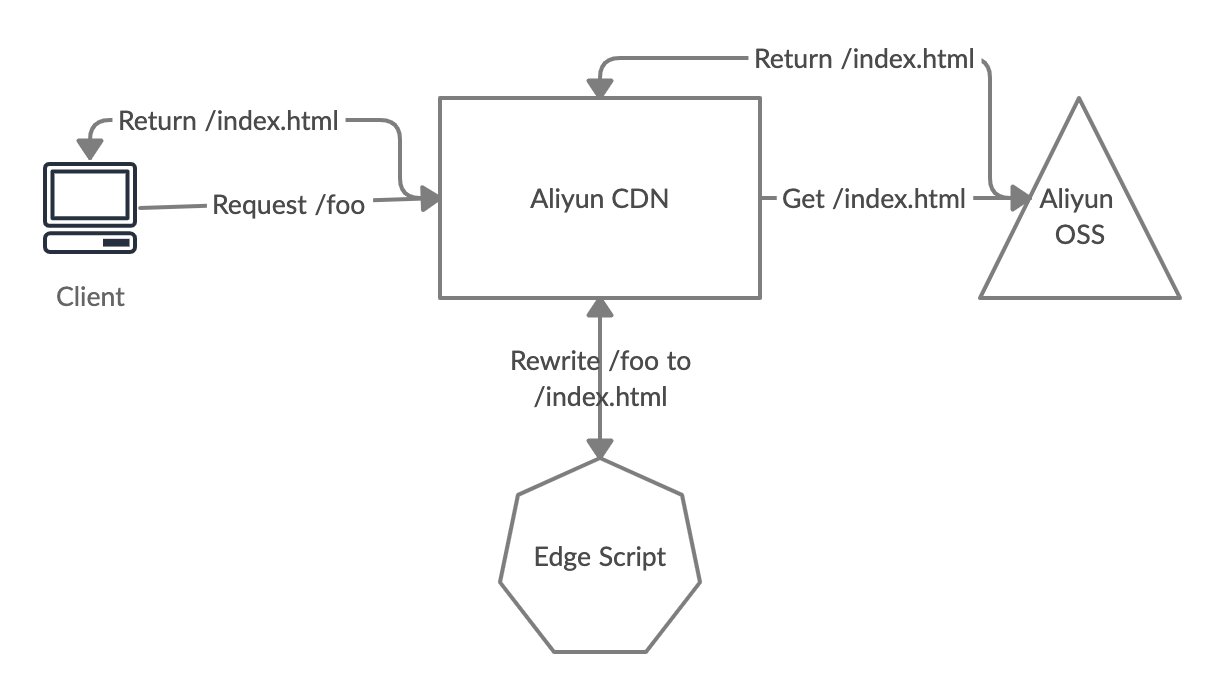
For URI starts with /feature, /hotfix or /release and have sub-directories: we will rewrite to sub-directory's index.html

For others: we will rewrite to /index.html

**S3**

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance.

**If go with Alicloud, what are the alternatives for CloudFront, Lambda and S3?**



We can use [Aliyun CDN](https://help.aliyun.com/product/27099.html)(CDN + [EdgeScript](https://help.aliyun.com/document_detail/126565.html)) + [Aliyun OSS](https://help.aliyun.com/product/31815.html) to implement the similar workflow.

|  |  |  |
| --- | --- | --- |
|  | AWS | Aliyun |
| Could Delivery Network | CloudFront | Aliyun CDN |
| Request URL Rewrite | Lambda | Aliyun CDN EdgeScript |
| Object Storage | S3 | Aliyun OSS |

We list the correspondence between the AWS and Aliyun. There are many alternative plan in Aliyun if we want to deploy in this platform.

**If go with IDC, what are the alternatives for CloudFront, Lambda and S3?**

|  |  |  |
| --- | --- | --- |
|  | AWS | IDC |
| Could Delivery Network | CloudFront | Squid-Cache |
| Request URL Rewrite | Lambda | Nginx |
| Object Storage | S3 | FastDFS/HDFS/TFS |

If we go with the IDC, there would be quite a lot of effort to set up all those things. Especially for the CDN part.

Logging and Monitoring

1. [Sentry](https://sentry.io/welcome/)

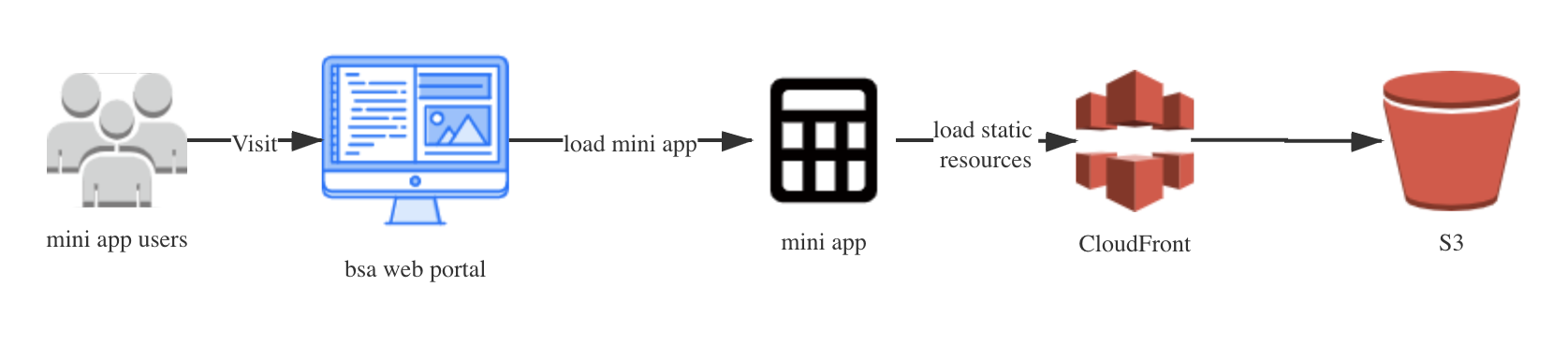
[JavaScript repo](https://github.com/getsentry/sentry-javascript/tree/master/packages/browser) github stars: 4.8k

[Npm package](https://www.npmjs.com/package/@sentry/browser) weekly downloads: 2,393,732

1. [Rollbar](https://rollbar.com/resources/)
2. [Datadog](https://www.datadoghq.com/)

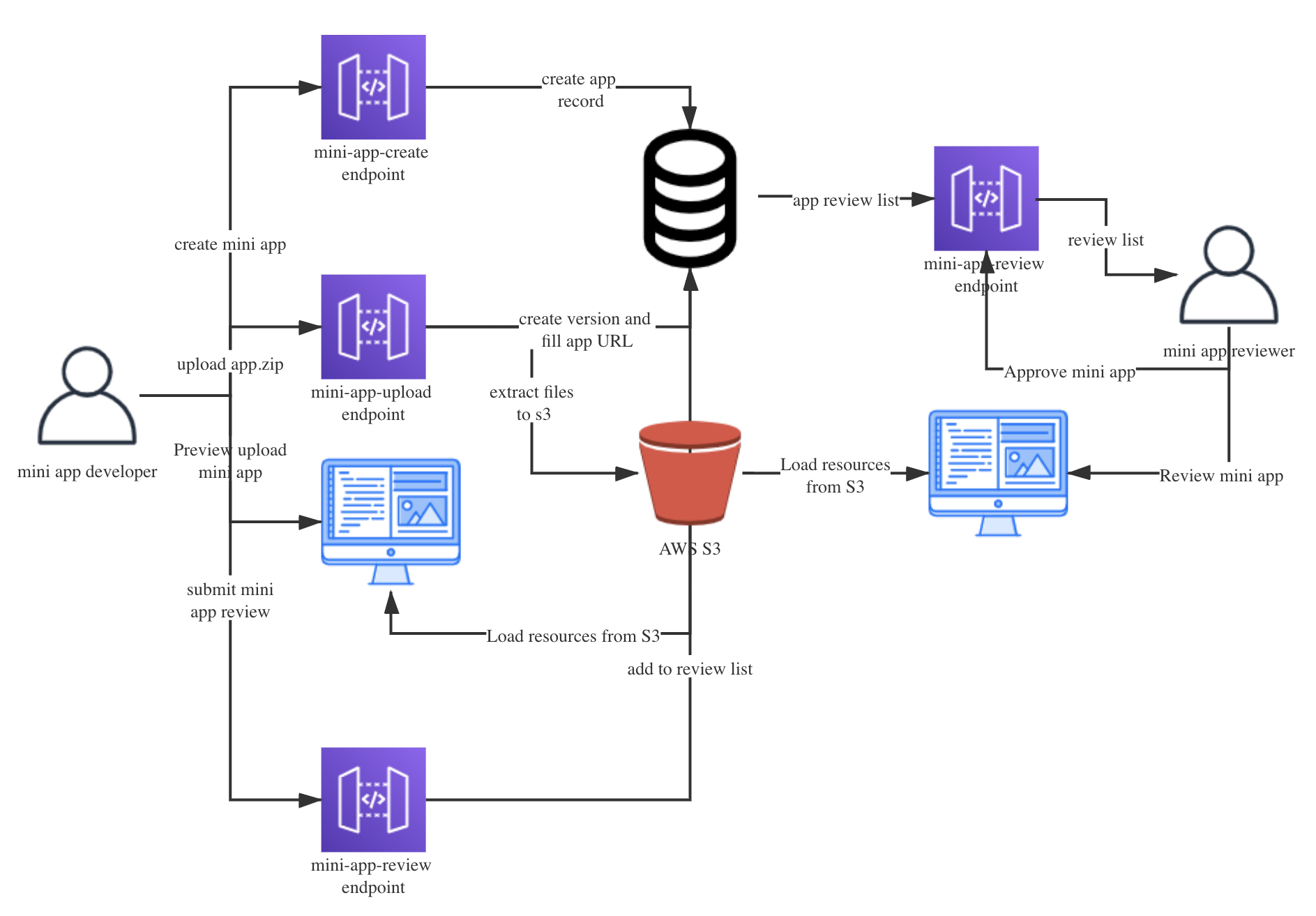
Mini App

User visit mini app



1. Users visit BSAweb portal page.
2. BSA web portal load mini app if is a mini app page
3. Browsers load static files from CloudFront
4. CloudFront will check cache if not exists will request from S3

Mini app upload & review



For mini app developer:

1. Developer create mini-app then backend will create database record for this mini-app
2. Developer upload app zip file then backend will create a new version use this zip file. Backend will extract this zip files to AWS S3 bucket. Then will generate a mini app link, send back to developer.
3. Developer will preview uploaded version after backend generate the link.
4. Developer will submit mini app review and then backend will update this version to ready to review.

For mini app reviewer:

1. Reviewer will get list of apps that pending for review, backend will return this review list by finding from db.
2. Reviewer will review the mini app. After that, they can approve this mini app.

Demo Web Project Repo – Gitlab

Demo project main frame: [web-starter](https://gitlab.com/REInventVenturesPteLtd/super-apps/web-starter), [online demo](http://dng2f90s38fdu.cloudfront.net/)

Demo mini app: [mini-app-test](https://gitlab.com/lin-reinvent/mini-app-test), [online iFrame demo](http://dng2f90s38fdu.cloudfront.net/iframe-page), [online qiankun demo](http://dng2f90s38fdu.cloudfront.net/qiankun)

Add a doc to tell how to configure CloudFront, S3, Lambda and DNS when first time to set up an environment for a new web project.?

For the configure about CloudFront, S3, Lambda, Please move to [this document](https://justoffice.sharepoint.com/:w:/s/REi-Engineer/ESjah_4YtlRKgUe-srjXbDYBmjX0HX5j0qkuYruCyKoiCA?e=EGh5Xv).

How about static image store?