# 

# 代码和文档管理规范

# 一、概述

本文档描述了产品研发项目的应用系统源代码及其相关开发资料的管理规范。

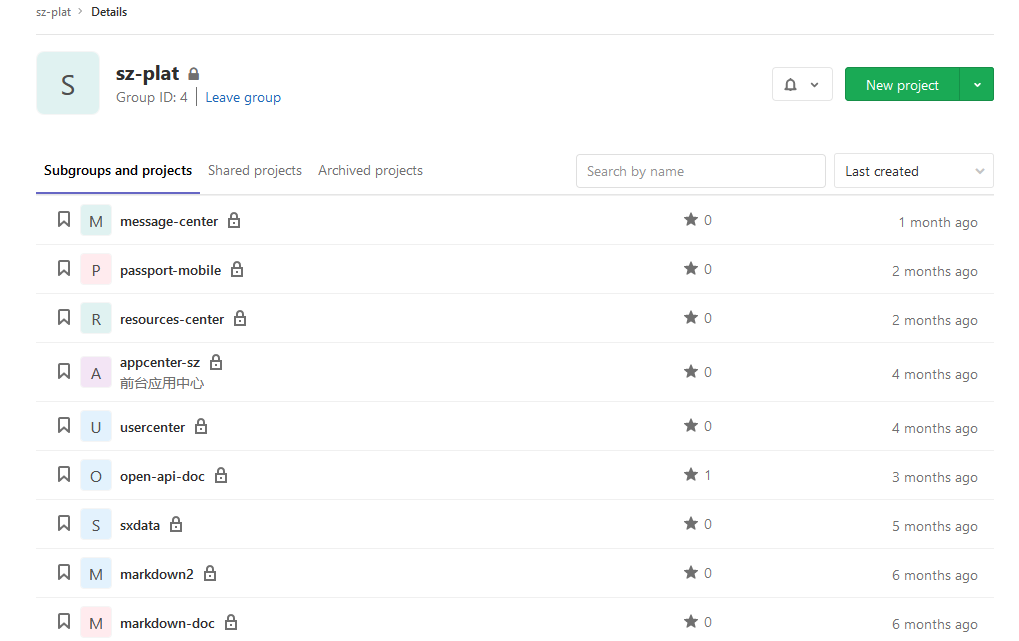
# 二、管理工具

我们采用gitlab（https://gitlab.hseduyun.net）管理代码及文档。

# 三、文件结构

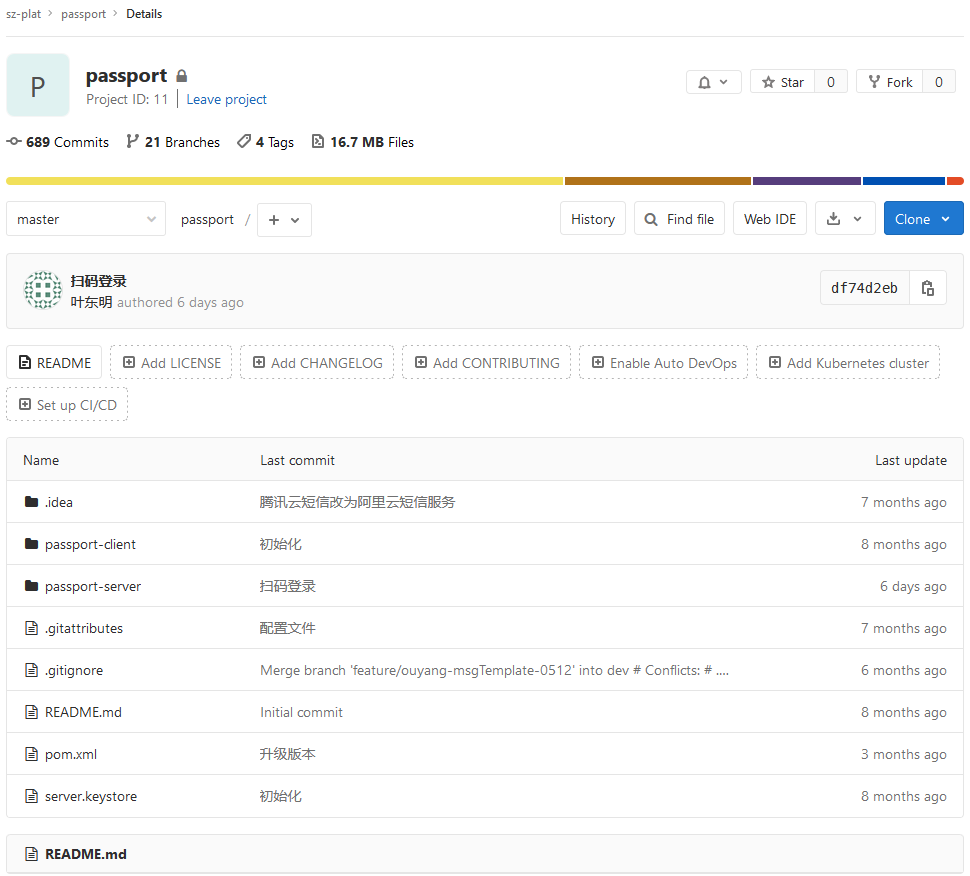
## （一）组目录

GitLab是基于组来控制用户对项目的访问的，一个组可以同时包含用户和项目，用户可以对自己组的项目进行访问，同一组的项目放在相同的组中，比如基础平台的项目全部放在基础平台的组中。如图：



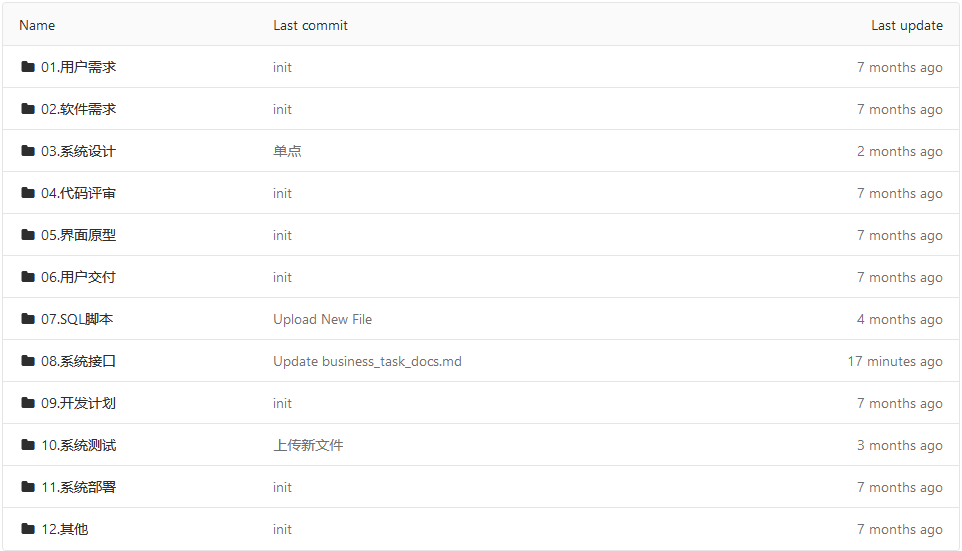
## （二）代码目录

代码目录，是管理代码分支和版本的地方，代码目录以项目名称命名，比如：授权服务器项目名称为passport则授权服务器代码目录名称就是passport。如：



## （三）文档目录

文档目录，是管理项目文档版本的地方，目前一个组目录里有一个文档目录,组目录里所有项目共用一个文档目录，文档目录里的内容如下：



# 四、命名规则

（一）组目录：

组目录以开发组名字命名，前面加sz+中划线，如基础平台组目录：sz-plat。

（二）代码目录

代码名称（代码目录）以项目名称英文单词或缩写组成，如有多个单词或缩写则单词之间用中划线连接，名称仅能包含小写英文字母和中划线。如授权服务的目录命名为passport是合格的，而PASSPORT，1PASSPORT，Passport\_App，PassportAPP等都是不合格的。

（三）文档目录

文档目录以组目录名（如上所示）+中划线+doc命名，如：sz-plat-doc。

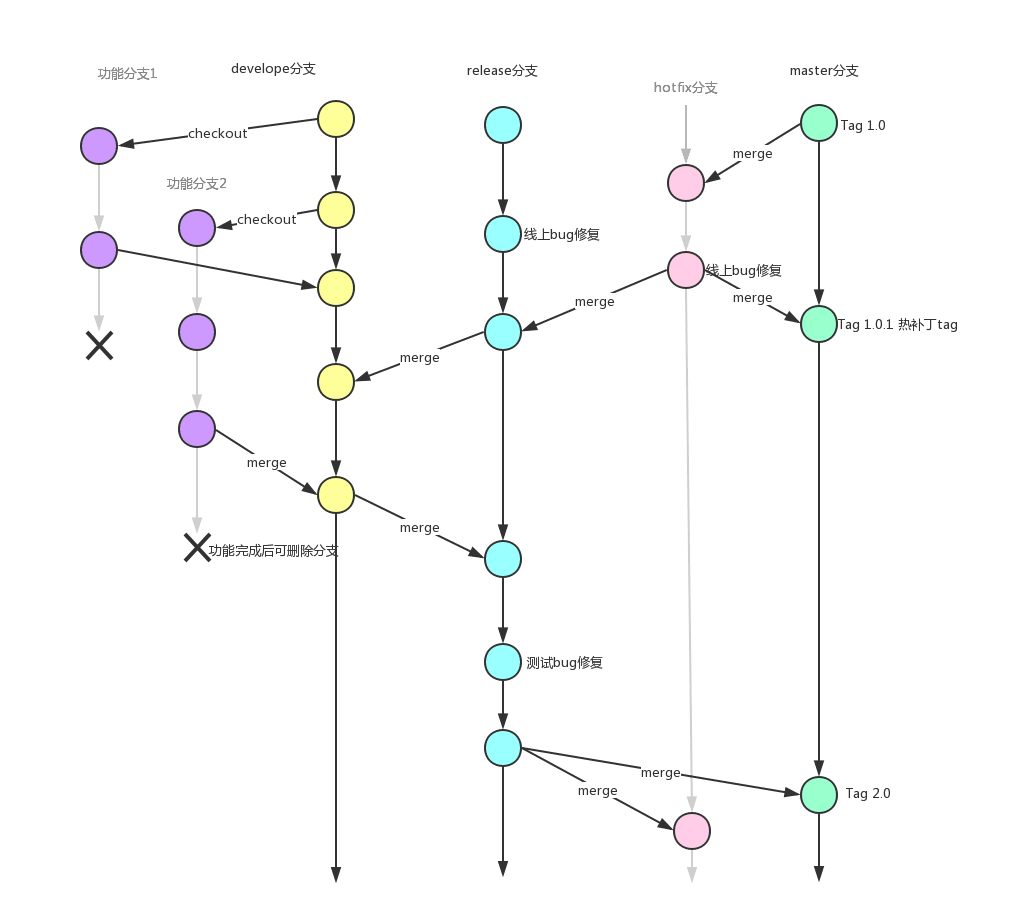
（四）文档

文档名用项目名+中划线+版本号+中划线+文档类型命名，如：passport-v1.2.0-用户需求说明书.doc、passport-v1.2.0-概要设计说明书。

（五）版本规则

小写的v＋三位版本号＋中划线＋版本描述（英文），中划线和描述是可选的。如： v1.3.1、v1.2.0-jsyd等。

# 五、分支 [Branches](https://gitlab.hseduyun.net/sz-plat/passport/-/branches)



## （一）develop分支

开发者在接到需求之后主要从事开发工作的分支。

## （二）release分支

测试在接到测试工作时主要使用的分支，在版本进入测试周期之后，会将该版本的测试代码从develop分支merge到release分支上，表示等待发布状态，此时将不会再添加新功能，只负责当前功能的bug修复。

## （三）master分支

当release分支上测试通过，表示代码随时能够发布时，会将代码merge到master分支，并且打上tag，至此，该版本封版，所有发现的bug均视为线上bug。

## （四）hotfix分支（次要）

该分支适用于存在热修复功能的开发系统，当发现线上bug之后，如果该bug可以使用hotfix进行修复的，就会在hotfix分支上修复，并且进行测试。测试通过后并入maste分支，并且打上tag号。

## （五）功能分支（次要）

功能分支是develop分支的辅助分支，将新功能在独立分支进行开发，开发完成后再并入develop分支。这样能够减少多人开发中，单个开发者由于新功能添加过程中修改代码对其他开发者造成的影响。

## （六）几个需注意的地方

1、master分支永远是稳定的代码，未经测试的代码不得合并入master分支。该分支只负责发布，不负责开发以及bug修复。

2、在版本开发完成之后，代码会合并入release分支进行测试。release分支的所有提交应该不包含新业务的开发，只包含当前需要发布版本的bug修复。

3、线上版本出现问题分为两类，一类能用hotfix修复，我们会使用hotfix分支修复后合并入master并且打上tag，并且还需要将hotfix分支的修复内容merge到release或者develop分支上（如果在release上修复，需要记得合并到develop分支中）。还有一部分无法进行hotfix修复，这些bug的修复分支可以在release上，也可以在develop上（如果在release上修复，需要记得合并到develop分支中）。

4、功能分支属于辅助分支，当新功能开发完毕并且合并入develop分支之后就逐渐失去作用，等到包含该功能的版本进入测试阶段，并入release之后，我们可以删除该分支，否则时间久了将会积累非常多的功能分支，反而不方便管理。

5、有时候我们需要测试接入对功能模块的单独测试，所以测试可能直接使用功能分支代码进行测试，这时候的bug修复就需要在功能分支上进行修复。

6、如果出现线上bug,需要在hotfix分支上修复（见5.4），版本号在原来版本上加一位，比如原来master封版后的版本号，为v1.2.0，出现线上bug修复合并后，打上tag号，版本号要变为v1.2.0.1。也就是只要出现4位数的版本号，就说明出现了线上bug，最后一位代表出现的线上bug次数。

7、开发提交测试后，测试经理从dev分支拉取release分支，测试通过后，测试经理出测试报告，通知产品经理，产品经理决定是否何时上线，并通知测试经理，测试经理合并release分支到master分支后，通知运维，运维执行上线操作。

附录：

文档目录说明

1、用户需求：产品经理上传用户需求说明书

2、软件需求：产品经理上传软件需求说明书

3、系统设计：架构师上传概要设计、详细设计

4、代码评审：研发经理上传代码走查报告

5、界面原型：产品经理上传产品原型

6、用户交付：产品经理上传安装手册、用户手册、版本说明书

7、SQL脚本：研发经理上传SQL脚本

8、系统接口：研发经理上传系统接口文档

9、开发计划：研发经理上传开发计划

10、系统测试：测试经理上传测试用例和测试报告

11、系统部署：运维上传运维流程规范、运维上线流程、运维操作手册、运维部署说明书

12、其他：产品生命周期产生的其他文档