ArchGov 系统部署说明书

前端视图组件的部署

1. 项目搭建和依赖安装

```
# 创建Vue项目
vue create your_project
cd your_project

# 安装Element-UI
vue add element

# 安装Echarts
vue add echarts
```

2. 数据交互和文件处理

在项目中使用Axios库，例如在Vue组件中：

```
// 安装axios
npm install axios

// 在组件中使用axios
import axios from 'axios';

// 发送GET请求
axios.get('/api/data')
  .then(response => {
    console.log(response.data);
  })
  .catch(error => {
    console.error(error);
  });

// 发送POST请求
axios.post('/api/upload', formData)
  .then(response => {
    console.log(response.data);
  })
  .catch(error => {
    console.error(error);
  });
```

3. 性能优化

使用Webpack进行代码打包和优化。在vue.config.js中配置：

```
module.exports = {
  configureWebpack: {
    performance: {
      hints: process.env.NODE_ENV === 'production' ? 'warning' : false,
      maxAssetSize: 1024000, // 设置文件大小阈值
    },
  },
};
```

4. 测试

使用Jest进行单元测试，创建测试文件例如your-component.spec.js：

```
import { mount } from '@vue/test-utils';
import YourComponent from '@/components/YourComponent.vue';

describe('YourComponent', () => {
  it('renders correctly', () => {
    const wrapper = mount(YourComponent);
    expect(wrapper.element).toMatchSnapshot();
  });
});
```

使用Cypress进行端到端测试，创建测试文件例如your-component.spec.js：

```
describe('YourComponent', () => {
  it('should do something', () => {
    cy.visit('/your-route');
    cy.get('button').click();
    // Add more test steps as needed
  });
});
```

5. 部署到云端

使用Nginx部署前端项目到云端。配置Nginx，将前端项目的build文件夹映射到Nginx的web目录，并配置反向代理。

```
server {
    listen 80;
    server_name your-domain.com;

    location / {
        root /path/to/your/project/build;
```

```
        index index.html;
        try_files $uri $uri/ /index.html;
    }

    # Add reverse proxy configuration if needed
    # location /api {
    #     proxy_pass http://backend-server;
    # }
}
```

提供访问地址和接口文档。

逻辑控制组件的部署

1. 微服务架构

使用Spring Boot和Spring Cloud搭建微服务架构。创建多个独立的服务，例如Auth Service、Analysis Service等。

2. 源码解析

使用ANTLR解析源码。在Java中引入ANTLR库，并编写解析器：

```
// 添加ANTLR依赖
implementation 'org.antlr:antlr4:4.9.2'

// 编写ANTLR解析器
// 示例代码略，根据实际需要编写ANTLR语法文件和解析器类
```

3. 版本历史分析

使用GitLab API获取提交信息。在Spring Boot服务中使用GitLab API：

```
// 使用GitLab API获取提交信息
// 示例代码略，需要根据实际情况编写获取和分析Git信息的代码
```

4. 大规模项目扫描

使用Spark扫描大规模项目。在服务中引入Spark库，编写扫描逻辑：

```
// 添加Spark依赖
implementation 'org.apache.spark:spark-core_2.12:3.2.0'

// 编写Spark扫描逻辑
// 示例代码略，根据实际情况编写Spark扫描任务
```

5. 测试

使用JUnit和Mockito进行测试。编写单元测试和集成测试：

```
// 添加JUnit和Mockito依赖
testImplementation 'junit:junit:4.13.2'
testImplementation 'org.mockito:mockito-core:3.11.2'

// 编写单元测试和集成测试
// 示例代码略，根据实际情况编写测试用例
```

6. 部署到云端

使用Docker和Kubernetes部署服务到云端。编写Dockerfile和Kubernetes部署文件：

```
# Dockerfile示例
FROM openjdk:11
COPY ./your-service.jar /app/
CMD ["java", "-jar", "/app/your-service.jar"]
```

```
# Kubernetes部署文件示例
apiVersion: apps/v1
kind: Deployment
metadata:
  name: your-service
spec:
  replicas: 3
  selector:
    matchLabels:
      app: your-service
  template:
    metadata:
      labels:
        app: your-service
    spec:
      containers:
        - name: your-service
          image: your-registry/your-service:latest
          ports:
            - containerPort: 8080
---
apiVersion: v1
kind: Service
metadata:
  name: your-service
spec:
  selector:
```

```
    app: your-service
  ports:
    - protocol: TCP
      port: 80
      targetPort: 8080
```

提供访问地址和接口文档。

数据存储组件的部署

1. 数据库设计

使用MySQL、Redis、MongoDB数据库。在数据库中创建表和索引：

```
-- 示例MySQL表创建语句
CREATE TABLE analysis_result (
    id INT PRIMARY KEY,
    code VARCHAR(255),
    result VARCHAR(255)
);

-- 示例Redis缓存设置
SET user_session:1234 "user_data";
```

2. 数据交互

使用MyBatis和Spring Data等工具简化数据交互。在Spring Boot服务中配置数据源和Repository：

```
// MyBatis配置
// 示例代码略，根据实际情况配置MyBatis

// Spring Data Repository
public interface AnalysisResultRepository extends JpaRepository<AnalysisResult,
Long> {
    // 示例查询方法
    AnalysisResult findByCode(String code);
}
```

3. 测试和迁移

使用DBUnit和Flyway进行测试和迁移。编写DBUnit测试

和Flyway脚本：

```
<!-- 示例DBUnit测试 -->
<dependency>
```

```
    <groupId>org.dbunit</groupId>
    <artifactId>dbunit</artifactId>
    <version>2.7.9</version>
    <scope>test</scope>
</dependency>

<!-- 示例Flyway脚本 -->
CREATE TABLE IF NOT EXISTS analysis_result (
    id INT PRIMARY KEY,
    code VARCHAR(255),
    result VARCHAR(255)
);
```

4. 部署到云端

使用云服务商提供的工具部署数据库到云端。在云平台创建数据库实例，获取连接地址：

```
# 云数据库配置示例
spring:
  datasource:
    url: jdbc:mysql://your-cloud-database:3306/your_database
    username: your_username
    password: your_password
```

提供访问地址和接口文档。