# 软件工程项目笔记

## 关于项目的部署

本软件项目使用Spring Boot作为底层启动框架，软件项目管理工具使用的是build.gradle。

其他有关的框架有：

1 使用Spring Data JPA作为持久层访问的框架，其特点是使用一系列@Annotaion作为实体的标记工具，用于控制数据库的同步生成。以及对数据持久层的访问。

2 使用Spring MVC作为web层的核心架构。页面模板引擎为Thymeleaf，其主要特点是可以快速显示Model中存储的信息，比较快速和方便。

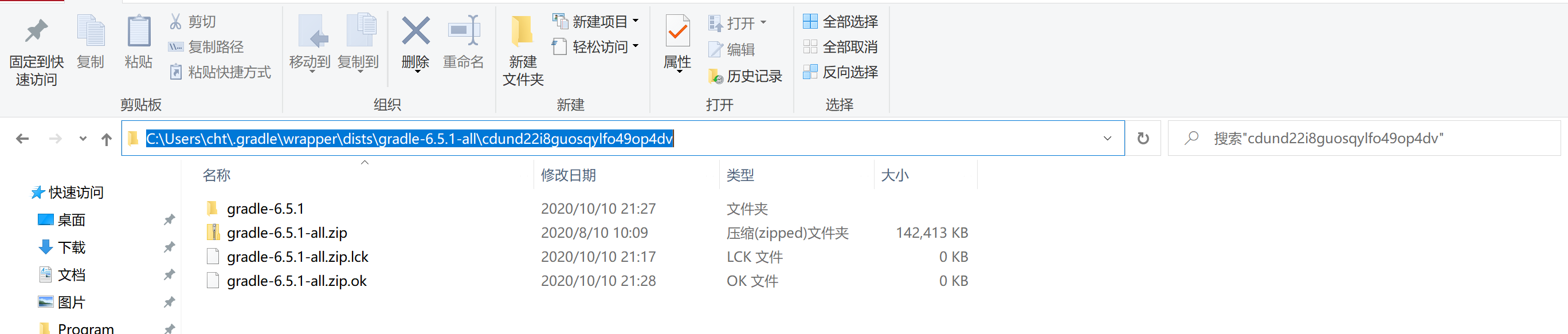
3 前端主要使用纯html和少量的js代码展示页面。

软件的部署分以下几部完成：

### 1 构建项目

用IntelliJ IDEA打开server文件夹，点击IDE右上角的build按钮进行项目构建。在构建过程中可能会出现下列问题。

a) 网络异常，无法下载gradle-6.5.1-all.zip。原因：gradle构建包有时候会由于网络问题无法下载，这个时候需要手动下载包到缓存目录。首先找到用户目录（常常为：c:/Users/${userName}），找到./gradle/wrapper/dists文件夹（如果没有则创建，然后），然后找到gradle-6.5.1-all/${base64}（为base64的一串码）/文件夹。然后将gradle-6.5.1-all.zip放到这个目录。



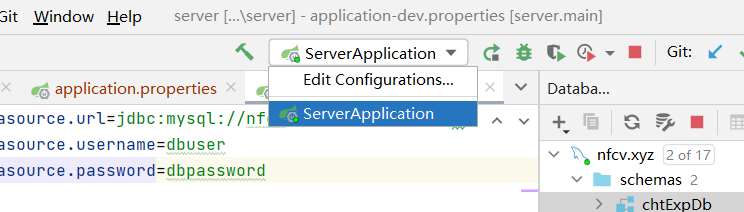
重新启动IDE，并开始测试。

### 2 配置数据库

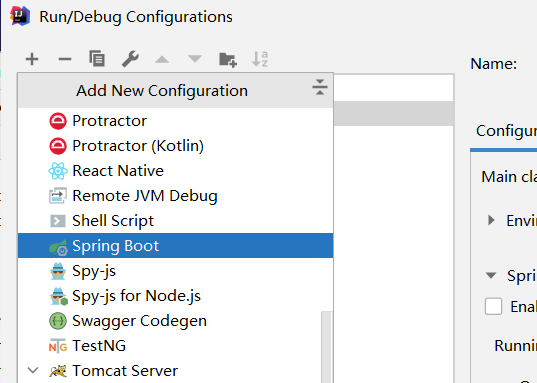
a) 在application.properties的同级目录中创建application-dev.properties，然后加入与数据库有关的配置项。例如：

|  |
| --- |
| spring.datasource.url=jdbc:mysql://nfcv.xyz:3306/chtLearning spring.datasource.username=dbuser spring.datasource.password=dbpassword |

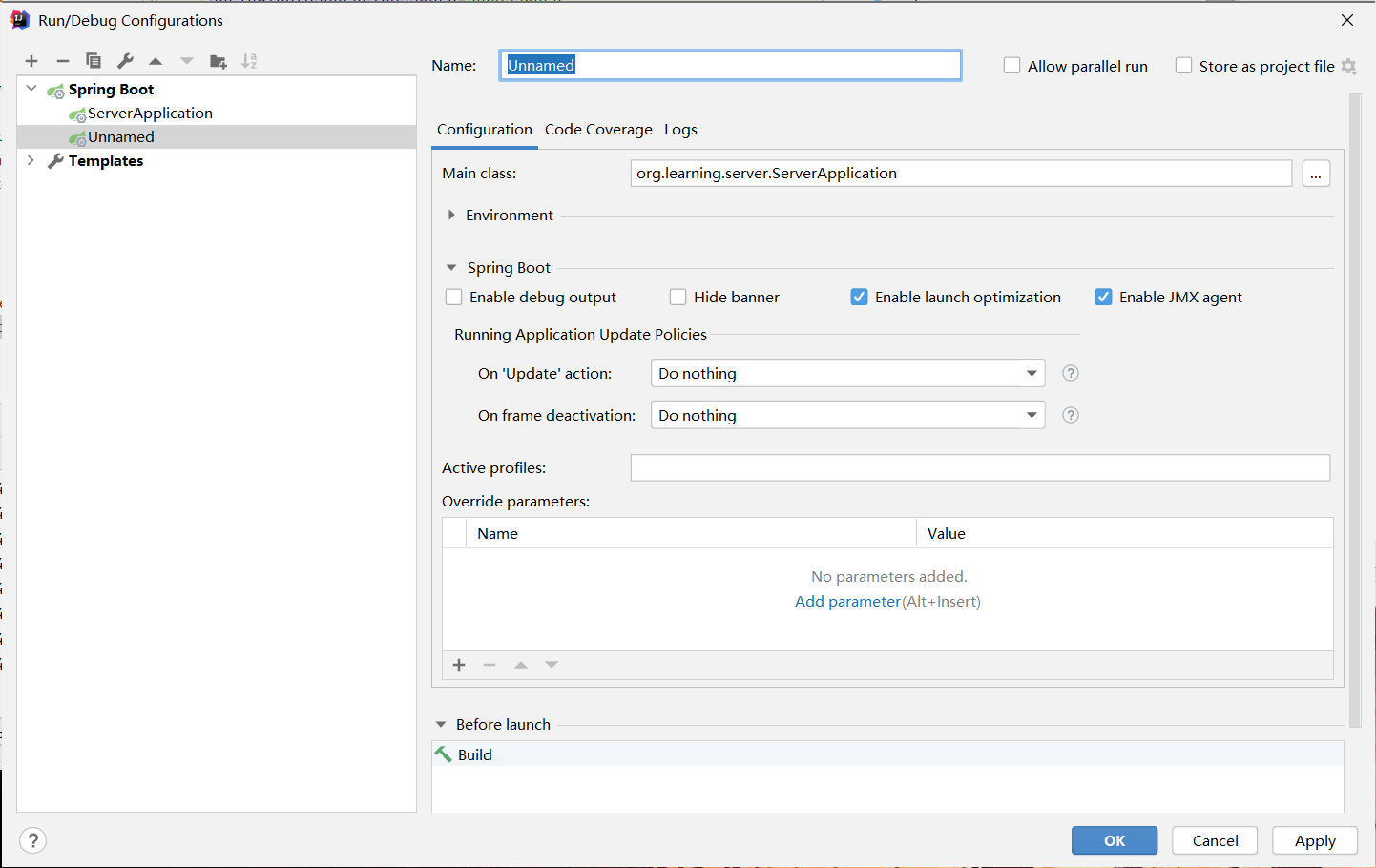
### 3 配置启动项



点击Edit Configuration…



找到Spring Boot



修改Name和Main class（org.learning.server.ServerApplication）。

## 关于build.gradle文件

build.gradle是部署与配置这个项目的核心。其中repositories指定软件依赖库的仓储。而dependencies指定软件的依赖。

在repositories中

|  |
| --- |
| maven **{** url 'https://maven.aliyun.com/repository/google' **}** maven **{** url 'https://maven.aliyun.com/repository/public' **}** |

两行制定了国内的maven源，主要是由于国际源常常由于网络的原因，连接不到。

dependencies中的各项即软件的依赖项，使用implementation/compile + 软件描述符的方式进行描述。其中软件描述符有3部分组成：groupName:archiveName:version。当软件需要依赖时，在其中添加依赖并build即可。

## 包的作用和说明

entity：存放与数据库实体相关的类。

repository：存放dao类型，其为一个接口，继承CrudRepository，从而实现数据库的Crud操作。

service：存放服务，业务的核心部分，一般能通过单元测试。其中实现类放在service.impl包中。

controller：控制器类，用于控制url->页面（数据）的跳转关系。

form：存放表单类型，一般用于存放可验证的表单类。

form.pattern：存放要用到的pattern信息

## 1.2 用户基础设置

使用到的教程：

1 如何进行表单验证（配合thymeleaf）<https://www.jianshu.com/p/48a4feab7541>

2 thymeleaf入门：<https://zhuanlan.zhihu.com/p/24988845?refer=dreawer>

数据库设计：请见实体entity.User

注册、登录：

使用到的Form：UserRegisterForm

使用到的其他类型：UserController，User，ActionResult，UserRepository，UserService

## 15周验收内容

1 组织使用树型结构进行管理。（使用回溯表节点进行管理）。分成多种树型的结构。

2 课程内容的设置参考Mooc。

3 体现组织之间管理的关系，例如上级的管理员可以对下级的节点进行管理。

4 课程资料库。课程。和课程开设的关系。

5 课程进度的记录，以及课程的管理等。

## 系统的结构设计

### 模块与功能

功能点：

1、（查找课程学习）能够查找当前进行中的**课程学习(CourseLearning)**，课程学习与课程，学员，架构节点关联（其中架构节点可以为空，表示该课程是学员自己主动学习的进程）。存在一个学员同时在不同的架构节点学习同一门课程的情况，不同的架构节点使用不同的**课程学习进度**。但可以通过课程学习进度同步操作同步课程的学习进度。

返回的信息：课程学习列表（全），在主页(/)上显示，接口/api/course\_learn/all

2、（智能查找）智能查找（现阶段由模糊匹配替换）该学员可以学习（已学习，公开的课程）的课程。并按照匹配度，热度（仅一个月内学习的人数），兴趣匹配度的函数进行排序。

返回的信息：课程列表（分页），在课程(/course)上显示，默认为按照热度，兴趣匹配度排序，接口/api/course/search?name=:{name}

3、（课程细节查询）用户从首页或者课程搜索界面进入，进行课程的学习，当没有学习的课程时，**显示订阅该课程按钮**。需要显示该课程的各个目录的信息，以及各个目录的完成度。

返回的信息：A、订阅的课程学习（如果没有则为空），以及对应的课程，还有**用户上次学习的位置**。接口/api/course/:{course\_id}&[course\_learn\_id=:{course\_learn\_id}]（这个为空表示用户是从课程搜索界面进入的）

B、如果用户已经订阅过该课程但从课程搜索界面中进入时，要求学员选择一个课程学习，**但是如果学员只有当前课程的课程学习实例，则自动跳转**。（CourseLearning）。可以通过查询/api/course\_learn/get?uid=:{uid}，点击后重新发送请求。

4、（课程章节查询）用户点击某一节（默认为第一个为学习的tab）或者某一节上某个资料的tab时，跳转到某一个资料的入口。并显示课程内容、课程学习进度等信息。

返回的信息：一个节内各个tab的信息，进度信息（如果是视频还需要携带上次最后观看的时间等），以及对应资源的信息，供后续调用。

接口/api/course/:{course\_id}/:{chapter\_id}?tabId=:{tabId}

5、（课程媒体获取）接4，通过媒体的查询获取对应的媒体资料。

返回的信息：媒体，接口/api/course/media/:{media\_id}

6、（管理员开设课程）