CS503全栈软件工程师直通车课程大纲

全栈软件工程师直通车项目旨在帮助学员在三个月的时间内从内而外全面提升能力，达到市场上对Full-Stack Engineer的招聘需求。

Project 1: Collaborative Online Judge System

【项目介绍】

通过本阶段项目，实现类似 Google Doc 的代码在线协作Online Judge System。课程将会带领大家从理论到实战，开发并部署 Node.js，搭建基于 Angular.js 的前端，并使用 Socket.io 完成多用户同步。通过部署并联通 MongoDB和Redis，利用使用 Nginx 做负载均衡，Docker Container 编译执行用户代码，以实现用户答案程序的判决。通过第一阶段项目，用户可以如同使用 Google Doc 一样相互合作共同编写代码，并浏览选择 coding problem；更可以针对指定的问题编程解答，系统将会编译运行用户代码并判决。这一项目在其他重要领域（如使用 Angular 框架构建前端，使用Node.js搭建 RESTful API 或使用 Docker 虚拟化技术等情形下）也有很好的拓展延伸性。

【学习成果】

1. 了解并掌握全栈的精髓语言Javascipt的运用，例如回调函数以及Promise；深入理解Chrome JavaScript引擎设计原理与细节
2. 学习并灵活运用当前流行的前端框架Angular的Architecture与模型是图设计，学会利用Angular CLI构建Single Page Application
3. 深入研究Node.js的运行原理
4. 掌握Server和Client之间传递数据的方式Restful API
5. 深入浅出学习MongoDB以及NoSQL的理论及其运用，掌握并进阶实战利用数据结构服务器Redis实现缓存设计
6. 研究主流协议WebSocket，对比各种Web Server的机制及流行趋势
7. 深入理解Nginx的反向代理功能
8. 掌握Docker的理论基础并学会使用docker编译并执行用户代码，实现隔离
9. 通过实例学习Reactive Programming以及流数据的处理；掌握Asynchronous Programming的理论与应用
10. 进一步构建技术壁垒，掌握Python语言基础；利用Python搭建backend server

Week 1 课程安排

【理论理解】

课程内容	课程要点
项目设计	• SNAKE原则 • 整体架构设计
Web 基础	• 了解构成网页的基本元素：HTML, CSS 核心知识点 • JavaScript重要应用
Node.js 的原理与运用	• Node.js 运行原理 • Node.js 特点与适用的环境 • Express 架构的特点与运用

【项目实战】

课程内容
设计系统结构
Chrome JavaScript引擎设计原理与细节
利用Angular CLI构建Single Page Application
动手搭建UI components，进一步掌握Angular框架
设计实现 Restful API
利用Node.js 实现 API Server，使用Node.js和Express搭建简单Server 并连接Client与Server

Week 2 课程安排

【理论理解】

课程内容	课程要点
Server和Client之间信息传递	• Restful API
WebSocket 原理与运用	• WebSocket 与 TCP • WebSocket 适用的场景 • Socket.io 的配置与适用
NoSql 数据库介绍	• SQL 数据库特点 • 为什么要使用 NoSQL 数据库 • 常见的 NoSQL 数据库
MongoDB 设计与实践	• MongoDB 基本原理 • 配置 MongoDB • 如何调用 MongoDB
Redis 设计与实践	• Redis 基本原理 • 配置 Redis • 如何调用 Redis

【项目实战】

课程内容
Reactive Programming以及流数据的处理
实现Restful API
利用MongoDB存储并获取数据
巩固WebSocket协议理论，并用socket.io实现多人实时在线编辑
实现 Redis 缓存设计

Week 3 课程安排

【理论理解】

课程内容	课程要点
Web Server	• 对比各种Web Server的机制及流行趋势
Nginx	• 反向代理 • 如何用Ngnix实现负载平衡
Docker 原理与运用	• Docker 原理 • Docker 与虚拟机的对比 • Docker 运用场景

【项目实战】

课程内容
利用Python搭建Backend Server
Nginx 搭建 Cluster
Docker 搭建快速迭代开发部署环境
系统性能测试与调优

Project 2: “今日头条”新闻挖掘与推荐系统

【项目介绍】

通过高强度的项目实战，深入强化全栈技能，突出强调系统整体架构设计，最终实现完整的 Web 前端与后端，健全的数据采集系统，和机器学习离线训练与线上预测系统。第二个项目我们将围绕另一项目“今日头条“ 新闻挖掘与推荐系统，带领大家开发部署Node.js Service，搭建基于 React 的前端并使用 Service-oriented 模式搭建 Python 后端。在这一阶段，同学们将熟练使用常用 Python 库，部署并联通 MongoDB和 RabbitMQ，利用网络爬虫抓取新闻。同时，我们将实现基于 TF-IDF 实现文档查重和基于 CNN 实现推荐系统两个重要功能。最后利用 TensorFlow 实现新闻 Topic Modeling，部署 TensorFlow Serving 提供在线预测。

【学习成果】

1. 理解网络爬虫与页面信息抓取流程与难点并学会利用网络爬虫抓取信息
2. 了解并掌握React 架构特性与基本原理，并搭建搭建基于 React 的前端
3. 熟悉服务导向型架构 (SOA) 的运用并学会设计系统
4. 熟悉消息队列的运用，部署并联通 RabbitMQ，使用 RabbitMQ 做负载均衡
5. 了解 Text Mining 基础，基于 TF-IDF 实现文档查重
6. 理解 Machine learning 基础原理与流程，了解常见机器学习问题与模型，掌握深度学习基础 与 TensorFlow 的运用, 并学会使用CNN实现推荐系统
7. 熟练掌握Python库，并使用RPC搭建Python后端

Week 4 课程安排

【理论理解】

课程内容	课程要点
Front-end 回顾	• 前端演进的各个阶段与代表技术 • 前端 MVC 框架的诞生 • Angular 与 React 对比与介绍 • Workshop: JavaScript 6.0 新特性
Nodejs 与 Express 回顾	• Nodejs 与传统多线程服务器区别 • NPM 社区 • Express 框架介绍

Workshop	• Workshop: JavaScript 6.0 新特性 ；React Basics

【项目实战】

课程内容
配置开发环境
分析 Web UI Component 关系
使用 create-react-app 创建 React app
自上而下编写各个 Component
使用 Rest API 与后端通信
使用 express 搭建 node.js 后端
使用 lodash 处理持续新闻读取中的抖动问题
实现用户注册与登录页面 UI

Week 5 课程安排

【理论理解】

课程内容	课程要点
NoSQL 数据库	• CAP 理论 • MongoDB 存储结构 • 使用 Python 来操作 MongoDB
消息队列	• 什么是消息队列 • 消息队列使用场景 • AMQP 协议模型与路由算法

	• RabbitMQ Python 库 pika 介绍
服务导向性（SOA）架构	• Amazon 与 SOA • SOA 架构优缺点
API 设计	• 什么是 API • RPC 与 REST • JSON RPC 与 XML RPC • JSON RPC Python 库 json-rpc 介绍 • API 设计原则

【项目实战】

课程内容
本地搭建部署 MongoDB，介绍 MongoDB shell 的使用
CloudAMQP 的设置与本地 Python 的联调
建立一个包含基本 RPC API 的 Backend Server
使用 Pylint 提高代码质量
使用 Json Web Token 实现本地 Authentication 与 Authorization

Week 6 课程安排

【理论理解】

课程内容	课程要点
Web Scraping	• 什么是爬虫，爬虫的应用 • XPath 介绍与基本语法 • Regular Expression 介绍与基本语法 • 使用 lxml 与 requests 实现一个简单爬虫 • User-agent 与 Proxy 的使用 • 多爬虫与消息队列的结合 • 如何应对反爬虫侦测
自然语言处理	• 自然语言处理基础 • Vector Space Model • TF-IDF • Topic Modeling
机器学习基础	• 基本概念与术语 • 监督学习基础 • 机器学习核心与 Workflow

【项目实战】

课程内容
CNN 新闻抓取示范
News API 调用示范
实现 News Monitor 用以监测最新新闻
实现 News Fetcher 用以新闻内容抓取
实现 News Deduper 用以新闻内容查重
使用消息队列实现完整 Pipeline

实现后端新闻的 Pagination
实现基于 Moving Average 的 Preference Model
实现Click Log Processor 收集用户点击事件并更新 Preference Model

Week 7 课程安排

【理论理解】

课程内容	课程要点
线性回归模型与机器学习原理	• Loss • Gradient Descent 与 Learning Rate • Generalization 与 Regularization • Representation
神经网络	• 神经网络简介 • 卷积神经网络基础 • 深度学习
TensorFlow 模型训练	• Pandas 的使用 • TensorFlow Programming Concept • 使用 TensorFlow 进行 Linear Regression 训练 • 如何选取特征 • 如何优化训练参数
Serving	• 部署 TensorFlow 模型 • 如何与 Python Backend 连接 • 如何自动化训练与动态更新线上模型

【项目实战】

课程内容
使用 Jupyter Notebook 做前期模型试验
尝试建立 CNN 模型用以新闻的 Topic Modeling
迁徙模型至工程代码实现线下训练
模型状态的存贮与还原
实现 TensorFlow Serving 服务
连接 Python Backend 与 TensorFlow Serving
自动化 Backfill 所有新闻进行 Topic Modeling
Logistic Regression 模型
kNN 模型

Project 3: “今日头条”iOS App

【项目介绍】

通过高强度的项目实战, 掌握Mobile Device的开发基础, 强化iOS开发技能, 并对比不同设计模式下系统设计以及其对应的编程方式。同学们将会使用多种iOS Widget、iOS Device硬件资源， 以及学会如何快速理解已有代码，相比于Service-Oriented Architecture在Object-Oriented Pattern下设计、实现扩展版iOS今日头条App。
第三个项目，首先通过讲解并实现一些小App帮助大家掌握Xcode与Swift基本使用方法, iOS程序Debug及Release流程。而后, 通过图形化界面以及AutoLayout建立今日头条App界面, 利用MapKit直观查看新闻源分布, 并利用Optical Character Recognition进行文字识别。在这个过程中, 同学们还将学会如何在OOP下如何使用“接口”；在iOS开发中进行Restful请求；对使用GPS, 相机等硬件进行用户许可请求；如何在iOS Device上部署离线ML Model；并最终熟练掌握iOS开发设计、构架、细节。

【学习成果】

1. 理解并掌握iOS构架的原理与使用
2. 掌握Swift 基础：语法， 数据类型；特性: Delegate和Protocol, Closure, Optional类型
3. MVC vs 函数式编程的异同
4. React Native(及类似语言) vs Swift的开发与应用
5. 实战Xcode storyboard设计、Autolayout、如何连接View和Controller
6. 如何使用iOS开发包管理Cocoapods以及第三方Package
7. iOS中处理手势Gesture，相机Camera，地理GPS，Restful请求与处理
8. 如何发布应用到App Store

Week 8 课程安排

【理论理解】

课程内容	课程要点
完成iOS App的工业流程及知识要点	• iOS趋势，现状 • 开发环境, 工具: Mac，Xcode 9.1+, iOS 11+ • iOS构架
熟悉并学习Swift基础功能的使用	• Swift基础 • Class与Struct

	• Method特点 • Data Structure • Optional, Lazy • Delegate用法举例
Xcode界面	• 界面， 创建Peoject,添加Class,Build, Debug • 用Xcode+Swift快速搭建iOS App
Button， Textfiled	• 掌握对button， textfiled等基础widgets的使用, 掌握如何使用Doc
理解MVC模式下各个模块直接交流的方式	• 登陆、注册ViewController原理， 设计 • 消息展示页面， 对TableView/Collection View设计和Controller的使用

【项目实战】

课程内容
计算器View的设计 • Widget(Button, Label)获取和使用方法 • Constraints, Autolayout 使用 • One design for all iOS devices
Controller, Model • View 和 Controller连接， 登录与注册的设计 • Swift语法特性初试MVC data flow • Swift 代码优化
实战Submit app到App Store
通过playground实战数据结构使用方法
实战segue的使用 • 实战delegate使用方法， 了解segue需要用delegate传递信息的原理 • Segue在注册、登陆跳转中的使用 • 新闻显示页面设计

Week 9 课程安排

【理论理解】

课程内容	课程要点
Cocoapods包管理简介	• 为什么使用包管理 • 如何使用Cocoapods • ProgressHUD 实现loading，success，fail状态表示
如何处理Restful请求	• Asynchronize vs Multithread • 如何处理response
iOS - GPS使用	• 对iOS设备地图，GPS的使用
React Native	• React Native(以及其他Native) vs Swift开发优劣

【项目实战】

课程内容
使用Alamofire对503 API进行请求：Authentication api & News api
iOS gesture 使用：晃动(shake)手机更换news
MapKit使用：A. 加载，刷新地图 B. 通过地理位置信息标准pins
GPS使用: 当前 GPS使用许可 plist 设置
React Native vs Swift: A. 对比优劣及使用情境 B. 举例React Native开发流程
拍照及图片提取，文字提取并展示
利用CoreML将Machine Learning Model加载至本地

Project 4: 跑步追踪系统

【项目介绍】

项目运用工业界最新的后端架构和开发技术，实现一个类似咕咚运动或者Nike+Running的实时位置模拟和追踪后台系统。 项目特别针对FLAG级别以及湾区独角兽公司的SDE面试中，必不可少的系统架构设计，微服务，以及分布式系统等问题，通过实例以及现场代码编写与讲解，让同学们融汇贯通微服务架构设计，云原生应用开发，SQL和NoSQL数据库，以及消息队列等核心知识。老师将手把手传授给学生工业界最新的后端架构和开发技术，同时有针对性的解释系统设计思维和编码技巧。通过完成本项目，学员将能真正掌握大规模分布式系统的设计与实现，在面试中脱颖而出。

这个课程的核心部分分为三点：

- Modern Java Application Development
- 系统设计和微服务架构
- 云原生应用开发

【学习成果】

1. 掌握Java后端开发核心知识，知道代码结构和最佳实践
2. 理解并掌握微服务原理及设计原则，能够熟练运用Spring Boot, Spring Data, JPA，内存数据库和MongoDB设计开发出微服务，并能进行配置管理和部署
3. 理解云原生应用的挑战，如何设计分布式环境下的微服务，能够熟练运用Spring Cloud和Netflix Open Source进行云原生应用开发
4. 理解并掌握服务监控和健康检查，知道运维挑战，能运用服务保险丝进行后端服务容错处理
5. 练习软件测试理论以及方法，能够熟练编写高质量代码，单元测试以及集成测试。

Week 10 课程安排

【理论理解】

课程内容	课程要点
Spring Boot 简介	• 什么是Spring Boot • 为什么要有Spring Boot • Spring Boot和传统Spring框架的对比
Spring Boot 重要功能	• 自动化框架配置 • 简化依赖管理

	• Embedded Tomcat • 生产环境特性 • 应用属性配置及管理
Spring Data简介	• 什么是Spring Data • 为什么使用Spring Data • 如何使用Spring Data • Spring Data进行数据访问的思路和最佳实践
Docker简介	• Docker带来的好处 • Docker应用

【项目实战】

课程内容
运用Vagrant安装本地开发环境 • xUbuntu • JDK • Git • Oh My Zsh • IntelliJ IDEA / Eclipse • Maven
使用Spring Boot快速实现REST API • 包依赖 • 框架配置 • 部署方式
使用In-Memory SQL DB • 引入依赖 • 启动内存数据库
编写REST API进行数据持久化 • 设计项目所需的实体类 • 通过REST API持久化实体类 • 通过REST API进行数据库查询

代码结构最佳实践 • 对项目进行面向对象分析和设计 • 使用Java Web application推荐的代码结构进行重构
Docker实战 • 使用Docker Compose • 启动Docker Container • 操作Docker Container

Week 11 课程安排

【理论理解】

课程内容	课程要点
消息中间件简介	• 主流的消息中间件 • RabbitMQ和Kafka对比
NoSQL数据库简介	• 什么是NoSQL数据库 • MongoDB简介
WebSocket简介	• 什么是WebSocket • WebSocket后端实现 • WebSocket前端实现
Spring Data REST简介	• Richardson's REST 成熟度模型 • 什么是Spring Data REST? • 为什么SpringDataREST提供了更简便的RESTAPI开发方法？ • 如何使用Spring Data REST?
系统设计	• 三层架构到多层架构设计要点与精髓 • 如何用代码体现架构 • 什么是好的REST API • 如何设计好的REST API • 如何进行抽象与解耦

【项目实战】

课程内容
实现大型分布式系统架构
使用Docker搭建消息中间件环境
使用Docker搭建NoSQL开发环境
使用Spring Data对MongoDB进行操作
实现location simulation service
实现location distribution service
实现location update service

Week 12 课程安排

【理论理解】

课程内容	课程要点
12 Factor Application简介	• 单一应用架构的现状及缺点 • 什么是12 Factor Application, 其中包含了哪些factors? • 为什么应用开发要遵循12 Factor原则
微服务架构简介	• 什么是微服务架构？ • 微服务带来的好处 • 为什么越来越多的企业采用微服务架构？ • 微服务在企业级应用的现状 • 微服务架构和其他架构的对比
Cloud Native Java开发简介	• Cloud Native Java背景介绍 • Cloud Native Java现状 • Cloud Native Java开发难点

	• Cloud Native Java开发技巧
Spring Cloud 简介	• 什么是Spring Cloud • Spring Cloud解决的问题 • Spring Cloud核心项目
Netflix Open Source 简介	• Netflix Open Source 包含了哪些项目 • Netflix Open Source 项目的优势 • Netflix Open Source 项目的应用
生产环境下软件开发及部署指南	• 什么是CI • 什么是CD • 如何做生产环境下部署
云环境下的运维	• 什么是运维 • 如何进行运维 • 云环境下运维的挑战

【项目实战】

课程内容
利用Netflix服务注册与发现以及服务保险丝对现有系统进行重构，增强分布式环境下的扩展性和容错性 • Service Registration • Service Discovery • Circuit Breaker • Edging Service
利用Spring Cloud进行Cloud Native Java 应用开发 • Spring Cloud Eureka • Spring Cloud Hystrix
使用Spring Boot Actuator进行运维，监控和健康检查 • Spring Boot Actuator API使用

Week 13 课程安排

【理论理解】

课程内容	课程要点
Software Quality简介	• 软件质量包含哪些方面？ • 软件质量的重要性 • 如何度量软件质量？
软件测试分类	• 软件测试有哪些方面？ • 静态测试 • 动态测试 • 白盒测试 • 黑盒测试 • 灰盒测试 • 回归测试 • 功能性测试 • 非功能性测试 • 性能测试
软件测试方法	• 单元测试 • 集成测试 • 端到端测试 • 性能测试
软件测试最佳实践	• 如何规划软件测试 • 如何设计软件测试 • 如何编写测试代码 • 如何验证测试结果
JUnit单元测试	• 什么是JUnit? • 为什么要使用JUnit? • 如何用JUnit编写单元测试? • 如何用JUnit验证测试结果？ • 其他单元测试框架
服务器端集成测试	• 什么是服务器端集成测试？

	• 集成测试框架简介 • 使用服务器端集成测试框架的好处 • 如何使用Spring进行服务器端集成测试？ • 如何验证mvc集成测试？
系统优化与扩展	• 系统瓶颈 • 系统优化途径 • 系统扩展方面 • 系统优化扩展常见问题

【项目实战】

课程内容
设计系统测试用例 • 设计测试用例 • 设计预期结果
使用JUnit编写单元测试 • 引入JUnit依赖 • 将JUnit集成进Spring • 运用Mockito进行Mock • 编写代码单元测试 • 执行检查结果并优化单元测试
使用Spring Testing框架编写服务器端集成测试 • 设计集成测试用例 • 引入Spring Testing依赖 • 编写集成测试代码 • 执行检查结果并优化集成测试

备注：课程大纲仅供参考，请以老师实际上课为准。