Fundamentals (12 Days)

This module aims at the basic digital literacy of the learner. After an overview of the technological basics of the programming languages and the JVM the learner will understand responsibilities of backend developers and get an intro to the basic tooling from the IDE over to the linux shell up to svc with git.

Topic	Content
Course Intro	information about course setup | technical setup of the laptops
Fundamentals	introduction to the linux terminal | simple bash commands
Versioning & Collaborating	Overview of Version Control Systems (VCS) | common basic workflows with git
Programming	HTML & CSS basics | basic principle of programming | overview of programming languages | IDE & Plugin
Introduction to Agile	scrum and kanban | comparison with classic project management

Java (72 Days)

This module is designed to introduce language and concepts. Basic programming skills will be combined with techniques like testing (TDD) and following coding standards. Practical skills like debugging and the usage/concepts of modern frameworks complete this module.

Topic	Content
Agile in Practise	Scrum for developers
Introduction	main entry point | primitive data types | use of variables | basic operators | commenting
Texts	string manipulation | StringBuilder
Dates	date and time types | formatting | manipulation | format and manipulation
Methods	Parameters |returns | scoping | reference variables
Statements and loops	if/else & switch | while / for loops | break & continue
Logical thinking	boolean type & operators
Collections	One- and multidimensional arrays array | manipulation | hashmap, lists etc.
Algorithmic thinking	Solving problems with algorithms using e.g. recursion and creating awareness of complexity / runtime
Debugging	How to use breakpoints and the debugger

OOP Basics	Classes | inheritance and objects
OOP Advanced	Constructors | abstract classes | interfaces | singleton | factories | polymorphism | casting | encapsulation
Testing	Concepts and types (unit,integration, functional, etc) | advantages of testing | practice TDD
Coding standards	idea behind standards | usage of linting tools
I/O	Read manipulate and write files
libraries	How to install and include libraries | Research and understand third party libraries
basic tools	Gradle | maven | webserver (tomcat)
exceptions	Principle of exceptions | syntax (try, catch) | read and interpret a stack trace
threads	threads |concurrency
functional programming	functional interface | streams
lambdas & functional interface	working with lambdas
module system & localisation	modules | localisation
Secure coding in SE	secure coding
Java new features	introduction to latest java version and new features

Databases (20 Days)

With the introduction of a modern persistence layer in the form of an RDB the learner will be able to read and write data, but also understand the basic principles, to avoid misuse and data inconsistency.

Topic	Content
Overview	what is a RDBMS | Tables | rows | columns | views | data types | SQL | comparison to NoSQL
Basic usage	install and create a PostgreSQL | use command line and desktop clients | read/write data with basic SQL
Usage in Java	connect to PostgreSQL using JDBC driver | Manipulate data as part of an Java program
Advanced SQL	more data types | joins | aggregate functions
consistency	transactions | locking | rollback
basic performance	costs of db operations | indices | n+1 query problems | slow queries

Framework + ORM (28 Days)

Frameworks standardize best practices and promote code reusability by providing libraries and components. They reduce time to bootstrap a new project while also doing away with boilerplate code.

Topic	Content
Spring & Spring Boot Introduction	What is the Spring Framework and its benefits | What is Spring Boot and its benefits
ORM	Hibernate
Core Concepts	Spring's IoC container | The Application Context | Dependency Injection | Beans | XML Configuration
Views & Templates	Application logic in Spring | Using Thymeleaf | Different requests & responses | organization of view in a project
Spring Web	Introduction | MVC Implementation | Controllers | Models | Services | Handling Errors
Forms	user input via forms | interaction frontend - backend | rendering forms | field types | data flow & validation
Spring Data JPA	Introduction | Repositories | Queries | Pagination and Sorting | Transactions
Spring Security	Introduction | URL Authorization | Login & Logout | Authentication | Spring Security with Thymeleaf
Spring Test	Introduction | Testing the Spring Application

APIs (14 Days)

Modern web services are not isolated but interact and communicate with each other. In this module the learner will consume and produce apis. Therefore create an understanding in clean service to service communication.

Topic	Content
basics	Concept of an API | API types | HTTP verbs and response codes | inspect popular examples
Consume REST API	Manually consume API (postman) | use requests library |error handling | understand 3rd party APIs
Create REST API	Create REST API | Basic authentication | Create documentation (OpenAPI & Swagger)

Cloud Services (14 Days)

After an introduction to classic server management and its challenges, the benefits of modern cloud services are explained to the learner. With this basic understanding, fundamental cloud services such as networks, computing and databases are introduced using AWS as an example.

Topic	Content
Intro	classical server management | Why use the cloud? | common cloud providers | IaaS vs PaaS vs SaaS
AWS overview	services overview
AWS - practical usage	use AWS console | create an account | use S3 to host a static page deploy to EC2 and RDS
GCP overview	GCP service overview | AppEngine | BigQuery | Cloud Storage

Final Project (20 Days)

The final project shows the learner's ability to apply and combine the acquired knowledge. Furthermore, the student will learn to structure bigger applications and experience all phases of a software project. The main focus of this module is to work in a team and collaboratively create and improve software.

Topic	Content
Project Management	form a team | brainstorm about project purpose | identify potential pitfalls
Agile methods	daily standups | split work into stories | sprint planning
Project implementation	working on project | code review | pair programming | create project presentation | documentation of project | deployment of project
Project presentation	slidedeck | project demonstration | presentation skills
Evaluation	detailed feedback by teachers and senior developers
Final Event	Feedback Session | Celebrating Success | Handing out DCI Certificate