**Detailed Backend Java Developer Roadmap: Phase-by-Phase Breakdown**

**This guide details all the must-cover topics for each phase of your 12-month backend Java developer journey. It’s organized for efficient self-study and mastery of essential skills, directly referencing content from your shared PDFs and the broader current industry landscape.**

**Phase 1: Foundation (Months 1-2)**

***Core Java & OOP***

| **Area** | **Topics to Cover** |
| --- | --- |
| Java Basics | Syntax, data types, variables & scopes, operators, conditionals (if, switch), loops, type casting, arrays, strings & methods, math operations, lifecycle of program java.pdf+1 |
| OOP Concepts | Classes & objects, attributes & methods, access specifiers, static/nested classes, inheritance, polymorphism, abstraction, encapsulation, interfaces, enums, record |
| Advanced OOP | Method overloading/overriding, static vs dynamic binding, lambda expressions, functional interfaces, annotations, packages, modules, initializer blocks, pass by value/reference java.pdf+1 |
| Collections | List, Set, Map, Queue, Stack, Dequeue, Iterator, Array vs ArrayList, generic collections, Optionals java.pdf+1 |
| Exception Handling | Try-catch-finally, custom exceptions, best practices |
| Build Tools | Maven, Gradle, Bazel (how to build/run/test/manage dependencies) [java.pdf](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/c9d43eee-ca51-4f91-a237-70aa1b749908/java.pdf) |
| IDE & Git | IDE setup (IntelliJ/Eclipse), Git fundamentals, workflow, version control, repo hosting (GitHub/GitLab/Bitbucket) backend.pdf+2 |

**Phase 2: Database Foundations (Month 2)**

***Relational and NoSQL Basics, Essential SQL***

| **Area** | **Topics to Cover** |
| --- | --- |
| SQL Syntax | SELECT, INSERT, UPDATE, DELETE, WHERE, GROUP BY, ORDER BY, HAVING sql.pdf+1 |
| Joins | INNER, LEFT, RIGHT, FULL OUTER, CROSS, SELF, subqueries (scalar/column/row/table) |
| Advanced SQL | Aggregate functions (SUM, COUNT, AVG, MIN, MAX), window functions, CTEs, indexes, constraints, performance queries |
| Database Design | Normalization (1NF/2NF/3NF), ACID, transactions (begin/commit/rollback), isolation, primary/foreign keys, DDL/DML/DCL, CRUD sql.pdf+1 |
| Database Tech | PostgreSQL, MySQL (focus), JDBC integration, schema design |
| NoSQL | Concepts, when to use, types: document (MongoDB), key-value (Redis), CAP theorem basics backend.pdf+1 |
| ORM | JPA, Hibernate (mapping, entities, relationships, persistence) java.pdf+2 |

**Phase 3: Spring Framework Core (Months 3-4)**

***Spring, DI, AOP, MVC, Core Boot Concepts***

| **Area** | **Topics to Cover** |
| --- | --- |
| Spring Core | IoC, Dependency Injection (constructor/setter/field), bean lifecycle, Spring configuration (XML/Java/Annotation) [spring-boot.pdf](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/42b5de7d-9f4e-44c3-aaae-f94836db59bb/spring-boot.pdf) |
| Spring AOP | Aspect-oriented concepts, advice, pointcuts, cross-cutting concerns |
| Boot Project | Spring Boot starters, auto-configuration, application.properties, loggers, environment profiles |
| Embedded Server | Tomcat/Jetty basics, running standalone jars [spring-boot.pdf](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/42b5de7d-9f4e-44c3-aaae-f94836db59bb/spring-boot.pdf) |
| Spring MVC | Controllers, REST endpoints, Mapping, Request/Response, validation, exception handling, static files |
| REST APIs | JSON/XML serialization, HTTP status, API documentation (Swagger/OpenAPI) |
| Testing Setup | JUnit, @SpringBootTest, testing strategies java.pdf+1 |

**Phase 4: Data Access & Persistence (Month 4)**

***Spring Data, JPA & Hibernate Advanced, Transactional Apps***

| **Area** | **Topics to Cover** |
| --- | --- |
| Spring Data JPA | Repositories, CRUD, custom queries, pagination, sorting, specifications, projections [spring-boot.pdf](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/42b5de7d-9f4e-44c3-aaae-f94836db59bb/spring-boot.pdf) |
| Hibernate | Entity relationships (OneToOne, OneToMany, ManyToOne, ManyToMany), fetch types, cascades |
| Entity Lifecycle | @Entity, @Table, @Id, EntityManager usage, persistence context |
| Transactions | ACID in Spring, @Transactional, isolation levels, rollback, savepoints |
| Performance Tuning | N+1 problem, Lazy vs Eager loading, caching, profiling |
| Database Integration | Multiple data sources, migrations (Flyway, Liquibase) |

**Phase 5: Security & Authentication (Month 5)**

***Secure Java Apps with Spring Security***

| **Area** | **Topics to Cover** |
| --- | --- |
| Authentication | Basic/Form, JWT, OAuth2, session vs token vs cookie, SAML, OpenID backend.pdf+1 |
| Authorization | Roles, permissions, method-level, URL security |
| Advanced Security | CSRF, CORS, HTTPS/SSL/TLS, password hashing (bcrypt, scrypt), server security |
| Security Coding | Spring Security config, custom user details, password encoders, integrating social logins |
| API Security | Secure endpoints, best practices, API key management, OAuth flows backend.pdf+1 |
| Security Testing | Vulnerability scans, penetration testing, reviewing OWASP top 10 |

**Phase 6: Testing (Month 6)**

***Comprehensive Java & Spring Testing***

| **Area** | **Topics to Cover** |
| --- | --- |
| Unit Testing | JUnit (5+), assertions, parameterized tests |
| Mocking | Mockito, @Mock, @MockBean, stubbing, spying |
| Integration Testing | Spring Boot integration (@SpringBootTest), TestContainers, database testing |
| Web/API Testing | MockMVC for controllers, REST Assured, Postman test suites |
| Test Automation | Test reporting, CI/CD integration, code coverage, static analysis |
| Behavior/Performance | Cucumber, JMeter, load tests |

**Phase 7: Microservices & Advanced Topics (Months 7-8)**

***Building Real Microservices with Spring and Cloud Tech***

| **Area** | **Topics to Cover** |
| --- | --- |
| Microservices Concepts | Principles & benefits, comparison to monolith, domain-driven design, twelve-factor app backend.pdf+1 |
| Service Discovery | Eureka, Consul [spring-boot.pdf](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/42b5de7d-9f4e-44c3-aaae-f94836db59bb/spring-boot.pdf) |
| API Gateway | Spring Cloud Gateway, route mapping, filters [spring-boot.pdf](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/42b5de7d-9f4e-44c3-aaae-f94836db59bb/spring-boot.pdf) |
| Distributed Config | Spring Cloud Config, central config server |
| Circuit Breakers | Resilience (Hystrix, Resilience4j), fallback logic, bulkheads |
| Messaging | RabbitMQ, Kafka (basics, publishing/consuming events) backend.pdf+1 |
| Tracing/Observability | Sleuth, Zipkin, distributed logging/tracing basics |
| Docker | Images, containers, networking, volumes, Docker Compose |
| CI/CD | Pipeline design, Jenkins/GitHub Actions for microservices |

**Phase 8: System Design & Architecture (Months 8-9)**

***Designing and Scaling Large Backend Systems***

| **Area** | **Topics to Cover** |
| --- | --- |
| Scalability | Horizontal vs vertical scaling, load balancers (Nginx, HAProxy), caching (Redis/Memcached) |
| High Availability | Replication, sharding, failover, master/slave |
| CAP Theorem | Understanding trade-offs, consistency/availability/partition tolerance backend.pdf+1 |
| Messaging Patterns | Event-driven, async processing, message queues (RabbitMQ, Kafka), pub-sub models |
| Design Patterns | Singleton, Factory, Observer, Strategy, Command, Repository, CQRS, Event Sourcing backend.pdf+1 |
| Architecture | Monolith vs microservices, SOA, DDD, Clean Architecture, layered architecture |
| System Design Scenarios | Estimate scale, API rate limiting, graceful degradation, throttling, circuit breakers |

**Phase 9: DevOps & Production (Month 10)**

***Deploying, Monitoring, and Running Java Apps in Production***

| **Area** | **Topics to Cover** |
| --- | --- |
| Version Control | Git workflow, branching, PRs, code reviews, CI/CD best practices backend.pdf+1 |
| Containers & Infra | Docker orchestration (Compose, Swarm), Kubernetes basics (pods, services), Helm |
| Deployment | Automated builds, rolling updates, blue-green/canary deploy, config management |
| Monitoring | App monitoring (Actuator, Prometheus, Grafana), log aggregation, metrics |
| Alerts & Telemetry | Application and infra metrics, setting up alert systems, health checks |
| Security Monitoring | Reviewing logs for security, SIEM basics |
| Performance Analysis | Profiling (JVisualVM, YourKit), JVM tuning, memory/leak analysis, GC logs |

**Phase 10: Advanced Topics & Specialization (Months 10-12)**

***Mastery and Niche Technologies***

| **Area** | **Topics to Cover** |
| --- | --- |
| Concurrency | Java threads, virtual threads, thread pool, concurrent collections, locks, memory model |
| JVM Internals | Tuning, garbage collection, performance optimization |
| NoSQL & Search | MongoDB (CRUD, queries, indexes), Neo4j (graph basics), Elasticsearch, Solr |
| Real-time Systems | WebSockets, Server Sent Events, polling strategies, InfluxDB, Cassandra basics |
| Event-Driven Systems | Kafka advanced (streams, partitions, consumer groups), event sourcing patterns |
| Project Portfolio | Showcase 3+ finished backend projects on GitHub, with documentation and API specifications |
| System Design Interviews | Practice design scenarios, prepare for DSA and system design interview questions |

**Project Guidance at Each Stage:**

| **Phase** | **Example Project** |
| --- | --- |
| Foundations | CLI/book management app, CRUD with basic Java & MySQL |
| Spring Core | Simple REST API (TODO app) with Spring Boot & database |
| Data Layer | Blog platform with Hibernate/Spring Data (users, posts, comments) |
| Security/Testing | User authentication system (JWT/OAuth2), tested endpoints |
| Microservices | Ecommerce + order management with Spring Cloud + Docker |
| System Design | Medium + notification service with robust scaling, caching, tracing |
| Production/DevOps | Deploy above apps to cloud (AWS/GCP), containerized, CI/CD pipelines |
| Specialization | Real-time chat app, search API, event-driven notification microservice |

**Additional Advice**

* **Use LeetCode, Edabit, HackerRank** to brush up DSA ().datastructures-and-algorithms.pdf+1
* **Make sure all code is versioned, tested, and shared** (GitHub).
* **Follow official documentation** especially Java (oracle.com), Spring (spring.io), and relevant database docs.
* **Build real-world projects** to consolidate your understanding.
* **Always prepare for system design and behavioral interview questions** for top jobs.

**This level of detail ensures you become a well-rounded backend engineer with top-notch proficiency in Java and Spring Boot. If you want topic-by-topic resources or week-by-week breakdowns for each phase, let me know!**

1. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/c9d43eee-ca51-4f91-a237-70aa1b749908/java.pdf>
2. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/5c59dfcd-563b-4b39-b2b8-461196293d22/SDE-Roadmap.pdf>
3. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/8bd645df-38eb-4910-ae35-58d435494b96/backend.pdf>
4. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/76e10120-4144-4e13-be43-65f4b978c4bd/sql.pdf>
5. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/42b5de7d-9f4e-44c3-aaae-f94836db59bb/spring-boot.pdf>
6. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/33c66093-bef0-41ed-928e-e17954b4fec5/system-design.pdf>
7. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/1b60fb26-3438-4c98-a727-cc9b0f961bcd/datastructures-and-algorithms.pdf>

**Detailed Backend Java Developer Roadmap: Phase-by-Phase Breakdown**

**This guide details all the must-cover topics for each phase of your 12-month backend Java developer journey. It’s organized for efficient self-study and mastery of essential skills, directly referencing content from your shared PDFs and the broader current industry landscape.**

**Phase 1: Foundation (Months 1-2)**

***Core Java & OOP***

| **Area** | **Topics to Cover** |
| --- | --- |
| Java Basics | Syntax, data types, variables & scopes, operators, conditionals (if, switch), loops, type casting, arrays, strings & methods, math operations, lifecycle of program java.pdf+1 |
| OOP Concepts | Classes & objects, attributes & methods, access specifiers, static/nested classes, inheritance, polymorphism, abstraction, encapsulation, interfaces, enums, record |
| Advanced OOP | Method overloading/overriding, static vs dynamic binding, lambda expressions, functional interfaces, annotations, packages, modules, initializer blocks, pass by value/reference java.pdf+1 |
| Collections | List, Set, Map, Queue, Stack, Dequeue, Iterator, Array vs ArrayList, generic collections, Optionals java.pdf+1 |
| Exception Handling | Try-catch-finally, custom exceptions, best practices |
| Build Tools | Maven, Gradle, Bazel (how to build/run/test/manage dependencies) [java.pdf](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/c9d43eee-ca51-4f91-a237-70aa1b749908/java.pdf) |
| IDE & Git | IDE setup (IntelliJ/Eclipse), Git fundamentals, workflow, version control, repo hosting (GitHub/GitLab/Bitbucket) backend.pdf+2 |

**Phase 2: Database Foundations (Month 2)**

***Relational and NoSQL Basics, Essential SQL***

| **Area** | **Topics to Cover** |
| --- | --- |
| SQL Syntax | SELECT, INSERT, UPDATE, DELETE, WHERE, GROUP BY, ORDER BY, HAVING sql.pdf+1 |
| Joins | INNER, LEFT, RIGHT, FULL OUTER, CROSS, SELF, subqueries (scalar/column/row/table) |
| Advanced SQL | Aggregate functions (SUM, COUNT, AVG, MIN, MAX), window functions, CTEs, indexes, constraints, performance queries |
| Database Design | Normalization (1NF/2NF/3NF), ACID, transactions (begin/commit/rollback), isolation, primary/foreign keys, DDL/DML/DCL, CRUD sql.pdf+1 |
| Database Tech | PostgreSQL, MySQL (focus), JDBC integration, schema design |
| NoSQL | Concepts, when to use, types: document (MongoDB), key-value (Redis), CAP theorem basics backend.pdf+1 |
| ORM | JPA, Hibernate (mapping, entities, relationships, persistence) java.pdf+2 |

**Phase 3: Spring Framework Core (Months 3-4)**

***Spring, DI, AOP, MVC, Core Boot Concepts***

| **Area** | **Topics to Cover** |
| --- | --- |
| Spring Core | IoC, Dependency Injection (constructor/setter/field), bean lifecycle, Spring configuration (XML/Java/Annotation) [spring-boot.pdf](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/42b5de7d-9f4e-44c3-aaae-f94836db59bb/spring-boot.pdf) |
| Spring AOP | Aspect-oriented concepts, advice, pointcuts, cross-cutting concerns |
| Boot Project | Spring Boot starters, auto-configuration, application.properties, loggers, environment profiles |
| Embedded Server | Tomcat/Jetty basics, running standalone jars [spring-boot.pdf](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/42b5de7d-9f4e-44c3-aaae-f94836db59bb/spring-boot.pdf) |
| Spring MVC | Controllers, REST endpoints, Mapping, Request/Response, validation, exception handling, static files |
| REST APIs | JSON/XML serialization, HTTP status, API documentation (Swagger/OpenAPI) |
| Testing Setup | JUnit, @SpringBootTest, testing strategies java.pdf+1 |

**Phase 4: Data Access & Persistence (Month 4)**

***Spring Data, JPA & Hibernate Advanced, Transactional Apps***

| **Area** | **Topics to Cover** |
| --- | --- |
| Spring Data JPA | Repositories, CRUD, custom queries, pagination, sorting, specifications, projections [spring-boot.pdf](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/42b5de7d-9f4e-44c3-aaae-f94836db59bb/spring-boot.pdf) |
| Hibernate | Entity relationships (OneToOne, OneToMany, ManyToOne, ManyToMany), fetch types, cascades |
| Entity Lifecycle | @Entity, @Table, @Id, EntityManager usage, persistence context |
| Transactions | ACID in Spring, @Transactional, isolation levels, rollback, savepoints |
| Performance Tuning | N+1 problem, Lazy vs Eager loading, caching, profiling |
| Database Integration | Multiple data sources, migrations (Flyway, Liquibase) |

**Phase 5: Security & Authentication (Month 5)**

***Secure Java Apps with Spring Security***

| **Area** | **Topics to Cover** |
| --- | --- |
| Authentication | Basic/Form, JWT, OAuth2, session vs token vs cookie, SAML, OpenID backend.pdf+1 |
| Authorization | Roles, permissions, method-level, URL security |
| Advanced Security | CSRF, CORS, HTTPS/SSL/TLS, password hashing (bcrypt, scrypt), server security |
| Security Coding | Spring Security config, custom user details, password encoders, integrating social logins |
| API Security | Secure endpoints, best practices, API key management, OAuth flows backend.pdf+1 |
| Security Testing | Vulnerability scans, penetration testing, reviewing OWASP top 10 |

**Phase 6: Testing (Month 6)**

***Comprehensive Java & Spring Testing***

| **Area** | **Topics to Cover** |
| --- | --- |
| Unit Testing | JUnit (5+), assertions, parameterized tests |
| Mocking | Mockito, @Mock, @MockBean, stubbing, spying |
| Integration Testing | Spring Boot integration (@SpringBootTest), TestContainers, database testing |
| Web/API Testing | MockMVC for controllers, REST Assured, Postman test suites |
| Test Automation | Test reporting, CI/CD integration, code coverage, static analysis |
| Behavior/Performance | Cucumber, JMeter, load tests |

**Phase 7: Microservices & Advanced Topics (Months 7-8)**

***Building Real Microservices with Spring and Cloud Tech***

| **Area** | **Topics to Cover** |
| --- | --- |
| Microservices Concepts | Principles & benefits, comparison to monolith, domain-driven design, twelve-factor app backend.pdf+1 |
| Service Discovery | Eureka, Consul [spring-boot.pdf](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/42b5de7d-9f4e-44c3-aaae-f94836db59bb/spring-boot.pdf) |
| API Gateway | Spring Cloud Gateway, route mapping, filters [spring-boot.pdf](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/42b5de7d-9f4e-44c3-aaae-f94836db59bb/spring-boot.pdf) |
| Distributed Config | Spring Cloud Config, central config server |
| Circuit Breakers | Resilience (Hystrix, Resilience4j), fallback logic, bulkheads |
| Messaging | RabbitMQ, Kafka (basics, publishing/consuming events) backend.pdf+1 |
| Tracing/Observability | Sleuth, Zipkin, distributed logging/tracing basics |
| Docker | Images, containers, networking, volumes, Docker Compose |
| CI/CD | Pipeline design, Jenkins/GitHub Actions for microservices |

**Phase 8: System Design & Architecture (Months 8-9)**

***Designing and Scaling Large Backend Systems***

| **Area** | **Topics to Cover** |
| --- | --- |
| Scalability | Horizontal vs vertical scaling, load balancers (Nginx, HAProxy), caching (Redis/Memcached) |
| High Availability | Replication, sharding, failover, master/slave |
| CAP Theorem | Understanding trade-offs, consistency/availability/partition tolerance backend.pdf+1 |
| Messaging Patterns | Event-driven, async processing, message queues (RabbitMQ, Kafka), pub-sub models |
| Design Patterns | Singleton, Factory, Observer, Strategy, Command, Repository, CQRS, Event Sourcing backend.pdf+1 |
| Architecture | Monolith vs microservices, SOA, DDD, Clean Architecture, layered architecture |
| System Design Scenarios | Estimate scale, API rate limiting, graceful degradation, throttling, circuit breakers |

**Phase 9: DevOps & Production (Month 10)**

***Deploying, Monitoring, and Running Java Apps in Production***

| **Area** | **Topics to Cover** |
| --- | --- |
| Version Control | Git workflow, branching, PRs, code reviews, CI/CD best practices backend.pdf+1 |
| Containers & Infra | Docker orchestration (Compose, Swarm), Kubernetes basics (pods, services), Helm |
| Deployment | Automated builds, rolling updates, blue-green/canary deploy, config management |
| Monitoring | App monitoring (Actuator, Prometheus, Grafana), log aggregation, metrics |
| Alerts & Telemetry | Application and infra metrics, setting up alert systems, health checks |
| Security Monitoring | Reviewing logs for security, SIEM basics |
| Performance Analysis | Profiling (JVisualVM, YourKit), JVM tuning, memory/leak analysis, GC logs |

**Phase 10: Advanced Topics & Specialization (Months 10-12)**

***Mastery and Niche Technologies***

| **Area** | **Topics to Cover** |
| --- | --- |
| Concurrency | Java threads, virtual threads, thread pool, concurrent collections, locks, memory model |
| JVM Internals | Tuning, garbage collection, performance optimization |
| NoSQL & Search | MongoDB (CRUD, queries, indexes), Neo4j (graph basics), Elasticsearch, Solr |
| Real-time Systems | WebSockets, Server Sent Events, polling strategies, InfluxDB, Cassandra basics |
| Event-Driven Systems | Kafka advanced (streams, partitions, consumer groups), event sourcing patterns |
| Project Portfolio | Showcase 3+ finished backend projects on GitHub, with documentation and API specifications |
| System Design Interviews | Practice design scenarios, prepare for DSA and system design interview questions |

**Project Guidance at Each Stage:**

| **Phase** | **Example Project** |
| --- | --- |
| Foundations | CLI/book management app, CRUD with basic Java & MySQL |
| Spring Core | Simple REST API (TODO app) with Spring Boot & database |
| Data Layer | Blog platform with Hibernate/Spring Data (users, posts, comments) |
| Security/Testing | User authentication system (JWT/OAuth2), tested endpoints |
| Microservices | Ecommerce + order management with Spring Cloud + Docker |
| System Design | Medium + notification service with robust scaling, caching, tracing |
| Production/DevOps | Deploy above apps to cloud (AWS/GCP), containerized, CI/CD pipelines |
| Specialization | Real-time chat app, search API, event-driven notification microservice |

**Additional Advice**

* **Use LeetCode, Edabit, HackerRank** to brush up DSA ().datastructures-and-algorithms.pdf+1
* **Make sure all code is versioned, tested, and shared** (GitHub).
* **Follow official documentation** especially Java (oracle.com), Spring (spring.io), and relevant database docs.
* **Build real-world projects** to consolidate your understanding.
* **Always prepare for system design and behavioral interview questions** for top jobs.

**This level of detail ensures you become a well-rounded backend engineer with top-notch proficiency in Java and Spring Boot. If you want topic-by-topic resources or week-by-week breakdowns for each phase, let me know!**

1. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/c9d43eee-ca51-4f91-a237-70aa1b749908/java.pdf>
2. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/5c59dfcd-563b-4b39-b2b8-461196293d22/SDE-Roadmap.pdf>
3. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/8bd645df-38eb-4910-ae35-58d435494b96/backend.pdf>
4. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/76e10120-4144-4e13-be43-65f4b978c4bd/sql.pdf>
5. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/42b5de7d-9f4e-44c3-aaae-f94836db59bb/spring-boot.pdf>
6. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/33c66093-bef0-41ed-928e-e17954b4fec5/system-design.pdf>
7. <https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/97765597/1b60fb26-3438-4c98-a727-cc9b0f961bcd/datastructures-and-algorithms.pdf>