Week 1 Class1 (docker, SQL, pg Admin 4)

B/S, browser/server, TCP/IP 协议
DBMS
What is sql?

SQL four category:
DDL, DML, DCL, TCL

Docker uses local OP only when local OP is linux.

Docker: 一种数据库类型只需要pull一次image，然后用这一个image可以创建很多container。
Docker: container 不只可以运行数据库, 还可以运行很多东西，比如ubuntu

QUESTIONS：

Oracle、DB2、Sybase、MS SQL Server、Informax 算是container吗？ 都用SQL操作吗？都可以在docker和pg admin 4运行吗？docker 和 pg admin 4是啥关系？

Docker new DB?
Pg admin 4 如何新建服务器？五大要素：localhost port（5432） 数据库 用户名 密码

TCL: what is transaction within data base?

CODE:

docker pull postgres (download an image for one database)
docker images (ls all images)
docker rmi mysql (remove image named mysql)
docker run --name postgres -e POSTGRES_DB=mypostgresDB -e POSTGRES_USER=admin -e POSTGRES_PASSWORD=123 -p 5431:5432 -d postgres (run a container)
(docker run --name dealerDB【container name 】 -e POSTGRES_DB=dealer【database name, used for pg Admin4】 -e POSTGRES_USER=admin -e POSTGRES_PASSWORD=password -p 5431【虚拟机的port,can be either 5431 or xxxxx(five-digit number,】:5432【虚拟机里应用的port, 即postgres】 -d 【持续运行 detached mode】 postgres)
docker inspect container_name (check user name and password)
docker container ls (check current running container, or use docker ps)
docker ps (ls running container)
docker ps -a
docker ps --filter status=exited (only list stopped container)
(Filter status: One of created, restarting, running, removing, paused, exited, or dead)
docker kill $(docker ps -q) (stop all containers)
docker stop $(docker ps -a -q) (stop all containers)
docker rm $(docker ps -a -q) (remove all containers)
docker rm $(docker ps -a -f status=exited -q) (remove all exited container)
docker restart my_container (re-run an existing but stopped container)

docker tag <old image name> <new image name>
docker rmi <image name>

Docker: built your own image: docker build -t <image name>

docker container create - - name <container name> -p 1234:5678 postgres

docker container stop myContainerdocker container start myContainer
docker rename myContainer myContainer2
docker rm myContainer2
docker exec -it my_db /bin/bash (enter a container, -i is interactive -t is allocate a pseudo, tty port)
(exit a container) exit

hub.docker.com search 'postgres'
docs.docker.com
w3schools.com SQL tutorial (home work)

How to use pg admin4:
Docker 起一个数据库之后，打开pg admin4，right click servers->create->server->give a server name in general (whatever you want)->connection->Host name=localhost->port=5432(value of -p in docker, or 32771)->data base=postgre（i.e. POSTGRES_DB）->user name=admin (specified by docker, default is postgres)->password (see docker)->save

Week1 Class2 (JDBC, maven, Intellij IDE, flyway)

Java -> JDBC -> DBMS

In terminal:
mvn clean (clean all .class files in target)
mvn compile (compile .java files and save compiled files in target)
mvn clean compile (chain command, first clean then compile)

```
static final String DB_URL = "jdbc:postgresql://localhost:5432/myproject_db";
static final String USER = "admin";
static final String PASS = "Training123!";
```

What is POJO? Plain old Java object

CRUD: create read update delete

DAO: data access object

Insert flyway

How to use maven.

QUESTIONS:
TCL: what is transaction within data base?
What does it mean "commits a Transaction."?
commit; 这句命令是在干嘛? (把数据写入磁盘，而不是在内存排队)
What is persistence layer? （和硬盘数据有关的部分，因为只有写入硬盘才是persistence）
What is the relationship between maven and persistence layer? （no）

```
[Mac:~]$docker run --name my_db -e POSTGRES_USER=admin -e POSTGRES_PASSWORD=12345 -p 5432 -d postgres
24c9acf07ef3d53ce33cefb323389e41c3da82e467d537a49b656e3ac452821c
[Mac:~]$docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
 NAMES
24c9acf07ef3        postgres            "docker-entrypoint.s…"   3 seconds ago       Up 2 seconds        0.0.0.0:32771->5432/tcp
 my_db
[Mac:~]$
```

Why is Port not what I specified? (when I use pg admin 4, I need to use 32771 for port.)

After adding <flyway.version>6.0.0-beta2</flyway.version> in properties section in pom.xml file. mvn flyway: clean still has error "No plugin found for prefix flyway …..”【run mvn under project directory】

How come we define a new class by using interface? (List a = new ArrayList();)

CODES:

How to mark folder as source in Intellij IDEA:
Set source folder, so IDE knows what file is source file: right click ArtifactId (top folder in the left column)->open model settings->click the folder the under main folder, then choose Tests->apply

maven create a new project: choose maven->check create from archetype->choose maven archetype-quick-start->click next->name groupID and artifactID->next->finish

Copy dependency in dependencies section (such as org.postgresql)
Copy plugin in plugins section (such as flyway)

(run mvn under the project directory)
mvn -version
mvn compile
mvn flyway: clean
mvn flyway: migrate

Create v1__create_tables.sql
Create v1.1__insert_data_into_tables.sql
mvn flyway:migrate

Week2 class1 (JUnit, Logger, Git)

OOPs concept (object oriented programming): Encapsulation, Inheritance, Polymorphism.

Set test folder Before using JUnit:
right click ArtifactId->open model settings->click the folder under test folder same as source name, then choose Tests->apply

JUnit @Before @After @Test @BeforeClass @AfterClass @Ignore
Test suite: 打包test
Test runner: only need to know (maven or IDE is a test runner)

assertEquals(val1,val2);

First Add 【**private** Logger **logger** = LoggerFactory.*getLogger*(**this**.getClass());】
Green hammer->Edit->VM options->add【-Dlogging.level.com.ascending=DEBUG】

cd ~/IdeaProjects/runtime
git init
git config --global user.email "lyu4@gmu.edu"
git config --global user.name "lnsdlszsqxxx"
git config --list

git add --all (put staging area for all files)
git add FILENAME (for a specific file, especially for a cached file, i.e. untracked file)
git commit -m "fitst commit" (-m message) (combine to master)
git remote add origin https://github.com/lnsdlszsqxxx/runtime.git (define a new variable, the value of which is the URL)
git remote show origin (show the value of origin)
git push origin master (save to github server, origin is like alias of the URL)

.gitignore

git clone https://github.com/lnsdlszsqxxx/runtime.git (get this UCL from the gethub website choose clone or download)

git checkout -b branch1 (create a new branch)
git branch -d branch_name (delete an existing branch, you need to be on another branch when you do this.)
git checkout master (shift to master branch, and IntelliJ IDE will shift automatically)
git branch (check current branch)
git status (check current branch)

git checkout master
git merge branchname (merge)
Tag

Use gitignore file to ignore some unwanted files.
git rm (will delete the file from hard disk)

git rm --cache (not be deleted from the disk, but rm from git tree, i.e. untrack)
git rm -r /target (-r for a file)
git rm --cached
git push origin master (after git rm --cached)

Some git undo commands:
git add FILENAME (undo git rm --cached FILENAME)
git reset (unstage the changes)
git reset --soft HEAD~1(undo commit)
git reset --hard HEAD (undo git rm, if not staged)
git reset --hard (undo local changes)

Some git stash commands:
git stash (stash current working area)
git stash list (list all stashes)
git stash clear (clear all stashes)
git stash apply <stash number> (stash number is also called stash index, shown by git stash list, the number in the bracket. The latest stash index is 0)

QUESTIONS:
What is staging area? (git)

Week2 Class2 (logger, review docker, Assert)

mvn -Dtest=TestAll test

mvn -Dtest=TestAll -Dorg.slf4j.simpleLogger.defaultLogLevel=debug test
(test TestAll class, debug is printing level, default level is info)

Assert.assertEquals() 【比较数值是否相等】
Assert.assertTrue()
Assert.assertNull()
Assert.assertFalse()
Assert.assertNotNull()
Assert.assertSame() 【比较地址是否一样】
Assert.assertNotSame()

```
String str1 = new String("ABC");
String str2 = new String("ABC");
String str3 = “ABC”;
String str4 = “ABC”;
```
【str1 and str2 use different address, two objects; str3 and str4 use the same address】

IDE can generate getters and setters by 【control N】

QUESTIONS:
Why no main method in test class? (JUnit has a driver inside.)

Slf4j v.s. Logger (slf4j is one of the APIs for logging)
https://www.vogella.com/tutorials/Logging/article.html#overview_loggerjava

Error for logger in TestEmployee.java?

Set logger level in Java code? (Yes, we can. But change code is not a good method.)

Fix the version error:
“command + ,”->build, execution, deployment->change java compiler to 1.8 version.

Interface Comparator has two methods compare and equals, why is it still a functional interface? (Or why can we still use Lambda expression?)
And Why we don’t need to implement all of the methods in comparator?
Answer: If an interface declares an abstract method **overriding one of the public methods of `java.lang.Object`**, that also **does *not* count** toward the interface's abstract method count since any implementation of the interface will have an implementation from `java.lang.Object` or elsewhere. Other methods can be applied to this rule are toString() and hashCode().

Week3, class1 (Hibernate)

Hibernate is one of the ORM tools: object relational mapping

Hibernate: Configuration (hibernate.cfg.xml file, including DB info & mapping class info), SessionFactory, Session（会话）, Transaction, Criteria, Query (HQL, Hibernate query language, oop). First-Level Cache, Second-Level Cache, (优化性能的缓冲区)

Factory class: a class producing new classes.

Session: similar to connection in JDBC.

Configuration file: hibernate.cfg.xml; hibernate.properties.

Persistence class: @Entity(tell hibernate this is a mapping class)(in front of class), @Table(name="table_name")(which table this class refers to)(in front of class), @Column(name="column_name")(in front of field, which is column name in many cases), @Id(mark primary key), @GeneratedValue(how to generate id)

id INTEGER NOT NULL default nextval('department_id_seq')
=> id SERIAL (so we can use @GeneratedValue(strategy = GenerationType.*IDENTITY*) in mapping class)
(Serial: small serial, serial, big serial)

SessionFactory: only created when start up, thread safe, each database has one SessionFactory and one configuration file.

In pom.xml: fluent-hibernate-core （so we can scan mapping classes automatically）

We can pass the DB info to hibernate by the following maven code:
mvn test -Dtest=com.ascending.training.repository.HibernateMappingTest\
-Ddatabase.driver=org.postgresql.Driver\
-Ddatabase.dialect=org.hibernate.dialect.PostgreSQL9Dialect\
-Ddatabase.url=jdbc:postgresql://localhost:5431/mypostgresDB\
-Ddatabase.user=admin\
-Ddatabase.password=123

Another way to passing the DB info is the following :
Add the following in the VM option (delete -ea) (no slash, *no space*):
-Ddatabase.driver=org.postgresql.Driver
-Ddatabase.dialect=org.hibernate.dialect.PostgreSQL9Dialect
-Ddatabase.url=jdbc:postgresql://localhost:5431/mypostgresDB
-Ddatabase.user=admin
-Ddatabase.password=123
【set default VM option: change the above parameters in the JUnit template, every script in the same project can share the same input, so you don't need to modify it for each run】

The following code in sessionFactory can get the upper information for hibernate:

```
String dbDriver = System.getProperty("database.driver"); //get configuration info from VM option
String dbDialect = System.getProperty("database.dialect");
String dbUrl = System.getProperty("database.url");
String dbUser = System.getProperty("database.user");
String dbPassword = System.getProperty("database.password");

//following code in sessionFactory generates configuration file, i.e. hibernate.cfg.xml
Configuration configuration = new Configuration();
```

```
Properties settings = new Properties();
settings.put(Environment.DRIVER, dbDriver);
settings.put(Environment.DIALECT, dbDialect);
settings.put(Environment.URL, dbUrl);
settings.put(Environment.USER, dbUser);
settings.put(Environment.PASS, dbPassword);
settings.put(Environment.SHOW_SQL, "true");
settings.put(Environment.CURRENT_SESSION_CONTEXT_CLASS, "thread");
configuration.setProperties(settings);

//scan mapping class and change configuration file accordingly
//so we need to add fluent-hibernate-core in pom.xml
//this info will also be added into the configuration file
//then SessionFactory will be able to build the sessionFactory field
EntityScanner.scanPackages(modelPackages).addTo(configuration);
```

QUESTIONS:

Where is hibernate configuration file (hibernate.cfg.xml)? (IntelliJ IDE will not generate such a file)

Week3 Class2 (Session, HQL)

Session is connection. Not thread safe 【you need to close it after using it】.

Transaction: a unit of serial operations or works. Achieve data consistency, and rollback incase something goes unexpected. (one of the operations goes wrong, all the done operations will be rolled back. Only when all operations are done right, the unit of works will get done.) (only needed when writing operation)

```
try(Session session = HibernateUtil.getSessionFactory().openSession()){}
```

try(A){}: A will close the opened session automatically, so programmers don't need to write final block.

Fire Alarm!

QUESTIONS:

Why do we have to rewrite hashCode and equals at the same time?
Why using hashCode is faster for searching?
Why cannot HashMap find the value with the same hashCode? (although two objects have the same dashcode and the same key value, if you don't over write equals(), they are compared by address but not the key value. That's why they are considered different.)

Week3 Random1

Criteria: equivalent to the WHERE part in "SELECT WHERE" statement. the condition.

Week3 Random2

Something should not be uploaded onto GitHub: Local configuration, .class files, .DS_Store,

Use gitignore file to ignore some unwanted files.
git rm (will delete the file)
git rm --cached (not be deleted from the disk, but rm from git tree, i.e. untracked)
git rm -r /target (-r for a file, delete from disk)

git rm --cached **/.DS_Store
git rm --cached

When we use gitHub, we will upload pom.xml for maven. But pom.xml file may have some information we don't want people to know, as gitHub is open source. The following example shows how to use place holders for the usr_name and password in flyway in the pom.xml file (note that the space in front is very tricky, delete some spaces and insert again may fix the problem):

```
<plugin>
  <groupId>org.flywaydb</groupId>
  <artifactId>flyway-maven-plugin</artifactId>
  <version>${flyway.version}</version>
  <configuration>
    <driver>org.postgresql.Driver</driver>
    <url>jdbc:postgresql://${db_url}</url>
    <user>${db_username}</user>
    <password>${db_password}</password>
    <schemas>
      <schema>public</schema>
    </schemas>
  </configuration>
</plugin>
```

To run flyway with variables, we need to initialize the variables in terminal when we run maven:
mvn flyway:clean -Ddb_url=localhost:5431/mypostgresDB -Ddb_username=admin -Ddb_password=123

Setup (seed data, in @Before part in JUnit) && teardown (set testing data to zero, in @After part in JUnit), testing database return to zero.
schema: data structure
flyway: Schema version control rool (what is schema).
Setup add record

How do you log in server through terminal?

READ:
To write README file in gitHub, use gitHub mark down. https://guides.github.com/features/mastering-markdown/

QUESTIONS:
why do we need to set data to zero before and after testing? (when for multiple testings, we need to make sure the data is the same primitive data. Testing may change the data.)

Week3 Random3

design pattern: Factory (such sessionFactory) v.s. Singleton (such as DAO)

QUESTIONS:
Under which circumstances will you run git add, git commit and git push, respectively?

Two colleagues pull a same master from gitHub. One merges his own branch on the master tree and pushes first. Can the other one push his word on the updated master tree?

The effect of markdown file can only be viewed from gitHub?

Do you edit markdown files under terminal or a markdown editor online?

After running git log, what is in the bracket?
commit da279d3a3dcafd49950f4b624c2f0a85fed7d836 (**origin/master**, **master**)
commit b6ac8e8d895051f4bc344d58b0fd991c3d068e23 (**HEAD -> branch2**, **origin/branch2**)

README.md cann't be merged to master?

Week 4 Random1

mutable and immutable （mutation 变异）
immutable object (String Object)

Lambda expression considers every local variable to be final or pseudo final.
Lambda expression is lazy.

Final classes, methods and members have different goals.
A final class cannot be extended by other classes. Depending on your design, this may be a desirable design property of this class. It does not mean that all members of this class are immutable by default! Logical immutability can be a side effect as in your example. If a class is final and all of its fields are as well, the object is *strongly immutable*.
Final methods mean that the method cannot be overridden in a subclass. This can be for example useful with the template method pattern where you do not want clients to override the template method instead of the hooks. This, again, has nothing to do with immutable state because it does not directly relate to a member being immutable.
Final fields is where immutablity comes really into play. If you placed the final keyword in front of `value` like `private final String value;`, you would make sure that a value could not be overriden no matter who extended what.

Week 4 Class 1 (Hibernate mapping relationship)

@Entity

parents (having foreign key) vs children (no foreign key).
When you add records in children table

In mapping class (model class), use Set but not List. This is a Hibernate bug.

Default FetchType:
@OneToOne: EAGER
@OneToMany: LAZY
@ManyToOne: EAGER
@ManyToMany: LAZY
(the rule is when there is on too many data, it is EAGER, because it doesn't matter.)

1:1 bidirectional
owning side, the side has foreign key.
@OneToOne
@JoinColumn(name="user_id",referencedColumnName="id") [user_id column name of the table having foreign key, id is the primary key of the children table]
private User usr;

inverse side, the side does not have foreign key.
@OneToOne(mappedBy="usr", cascade=CascadeType.REMOVE)
private Address address;

1:n
@OneToMany(mappedBy="user", cascade=CascadeType.REMOVE, fetch=FetchType.LAZY)
private Set<Account> account;
(LAZY: get data when the data is needed, condition: the session should keep open)
(EAGER: get all of the data when defined.)

n:1
@ManyToOne(fetch=FetchType.LAZY)
@JoinColumn(name="USER_ID", referencedColumnName="id")
private User user;

m:n
@ManyToMany(mappedBy="employees", fetch=FetchType.LAZY)
private Set<Project> projects;

@JoinTable(name="Employee_Project",
joinColumns={@JoinColumn(name="project_id"),
inverseJoinColumns={@JoinColumn(name="employee_id")}}
private Set<Employee> employees;

Homework:
Test all DAO methods.

QUESTIONS:
How to rename employee model?

Week 4 Class2 (Hibernate remaining, restful web service, DI)

The relationship between tables is determined by the data base.

some column, such as user name: unique key.

left join vs left join fetch 【**when using join for three or more levels, use Set but not List in the mapping class**】
fetch: record
no fetch: Array object

What is Set? (In Hibernate, List is not good for two level children, Set is good for multiple levels.)

Owning side (foreign key side):
dept = departmentDao.getbyName("CS"); [get the department first, you need to tell Hibernate the relationship between employee and department. So Hibernate knows where to put the new employee record.]
e.setDepartment(dept);
session.save(e);

VM option (default): change the JUnit template

URI vs URL
URI > URL

HTTP methods: GET, POST, DELETE, PUT, PATCH
Representation: JSON(commonly used), XML, Text
Idempotence: PUT, DELETE, GET, HEAD, OPTIONS, TRACE
POST is not idempotent.

DI: Dependency Injection
a design pattern that removes the dependency initialization from compile-time to runtime.

mock object.

QUESTION:
Set? List? Stream?

Week4 Random2 (MultiThreading)

MultiThreading:
synchronized: both method and code block
lock: volatile (don't copy to local memory, but use the original memory), variable

difference between Runnable vs Callable
method name: run() call()
return type: void for run(), Future for call()
exception handling is different: callable can throw exception to upper level, runnable cannot.

to run a thread,
executor.execute: only for runnable task
executor.submit: both runnable and callable task.

threadpool: how many thread in the pool.

QUESTIONS:
Why is there no "catch" after "try" for some getters in the DepartmentDaoImpl files?
【exception is handled in HibernateUtil.java】

FetchType can be used on both sides, owning side and inverse side? 【yes】

Print all the elements in List? 【System.out.println(aList);】

How to write Assertion for getDepartmentsTest? (when there are too many records)

cascade=CascadeType.REMOVE is only used for @OneToOne and @OneToMany relationship and only for the inverse side?

ERROR: update or delete on table "students" violates foreign key constraint "account_student_fk" on table "accounts"
Detail: Key (id)=(1) is still referenced from table "accounts". 【REMOVE is used by Hibernate not by SQL. Use session.delete(department) works】

When it comes to three or more levels “join fetch”, we need to use Set but not List. Why do you only change the Lists in model classes not in repository classes? 【only alter List to Set in mapping classes.】

Why modify List to Set is wrong in repository classes, conflicting with “return query.list()”? 【see above】

Why can’t I use the second statement? 【Set doesn’t have get()】
-logger.info(department.getStudents().stream().findFirst().get().getAccounts().toString());
-logger.info(department.getStudents().get(0) .getAccounts().toString());

Week 5, Random 1 (transaction)

Hibernate transaction.
transaction = session.beginTransaction();
.
.
.
transaction.commit();

In complicated situation, the above transaction is not enough. For example, there are two methods in the code. Each one has a transaction. In some cases, the operation you want to rollback is complicated. Sometimes you want both of them to rollback, or just some of them rollback. You need to use some annotations in Spring Frame work, such as:
nested transaction, never, mandatory, support, ...

Week 5, Class1(service layer, Spring boot, DI, controller)

service layer: design pattern for modularization (模块化). For small project service layer is not necessary.
annotations:
@Autowired: DI feature
@Qualifier: solve conflicts, in case one interface implemented by two or more classes.

***don’t use slf4j API when using spring frame work, they conflict* (Spring has logger built in)**

Set spring logger level in the template (doesn’t work for spring test):
-Dlogging.level.org.springframework=INFO
-Dlogging.level.com.ascending=TRACE

controller: this term is used in spring.

Usually add @JsonIgnore in front of @ManyToOne.

QUESTIONS:
What is DI?
Why is Logger still right when slf4j commended out?

Try to use xml file in the hibernate, don’t generate the configuration file automatically.

Try to use Spring only but not Spring boot, learn how to write the configuration files. So understand why Spring boot is so convenient?

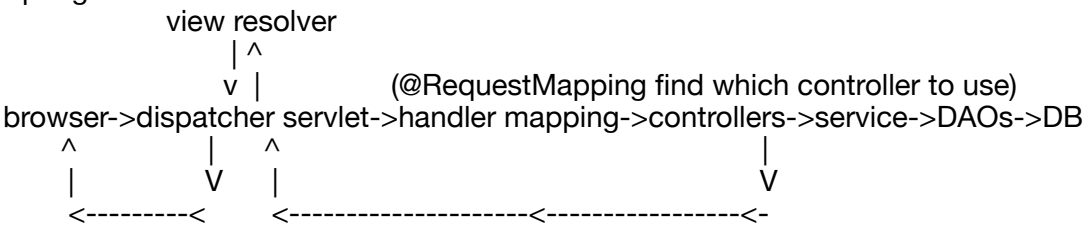
Week 5, Class 2 (Controller, Postman)

postman: can add data into the database. (browser can only review the result)

Tomcat: a web server, like apache.

relationship between server, Tomcat, spring, dispatcher servlet,
Server>tomcat>spring

Spring MVC architecture:



Four methods for sending data from front end to the back end:

1. path parameters (for RESTful web server, we use this method): www.google.com/login/dwang/123
2. query string (or request parameters), such as: www.google.com/login?username=dwang&password=123, in this example, we send two parameters username and password 【don't add quotes in the URL for the values. Everything is URL is String】
3. Header (secure)
4. body (for large amount of data)

Accordingly, there are four ways getting data from the request

1. @PathVaraible
2. @RequestParam. method_name(@RequestParam Map<String, String> allParams) get all parameters; method_name(@RequestParam(name="param") String pa)), when input, you should use param, but not pa. This is rename.
3. @RequestHeader
4. @RequestBody

QUESTIONS:

@JsonIgnore -> Do we have @XmlIgnore? 【XML doesn't need this】

Week 5, Class 2 residual

Shortcut for @RequestMapping(method = RequestMethod.*GET*), so we can ignore the method initialization in the parentheses:
@GetMapping
@PostMapping
@PutMapping
@DeleteMapping
@PatchMapping

```
@GetMapping(value = "/{deptName}", produces = {MediaType.APPLICATION_JSON_VALUE,
MediaType.APPLICATION_XML_VALUE})
public Department getDepartmentByName(@PathVariable(name = "deptName") String
deptName1)
```

- “{}” means that the deptName is a varaible, but not a path.
- If the input variable name (deptName) is different from the method variable name (deptName1), you need to use (name = “deptName”).

```
@PostMapping(value = "", consumes = {MediaType.APPLICATION_JSON_VALUE})
```

consumes: server side needs what data type. (Spring will convert the Json type into a java object), “{}” after consumes allows multiple data type.
if you want to use XML, please add the dependency in your pom.xml

```
<dependency>
  <groupId>com.fasterxml.jackson.dataformat</groupId>
  <artifactId>jackson-dataformat-xml</artifactId>
</dependency>
```

QUESTIONS:

How to input two records from the @RequestBody block? 【In json format, use[{department1}, {department2}]】

How many methods would we include in the controller? Only write four methods for each verb, respectively? 【No, it depends on your API functions】

How to input @RequestHeader from postman? 【similar to @RequestBody】

In our example, it seems that the service layer is the same as DAOs. How to write service layer in reality? 【this is only for learning purpose. In reality, service layer is based on your business logic.】

We cannot write test class for controllers. Only test controllers in Postman? Is there anything like TestSuite in Postman? 【We can use unit test and test suite in Java. Postman is not convenient for too many controller test.】

Homework: write four input requests. (what is @PatchMapping?)

What is the difference between @PatchMapping and @PutMapping? 【For patch, you can only change part of a object; for put, you need to provide all the information】

Week 5, Random 2 (DI, Spring mechanism)

Java is coffee, Java Bean is used to make coffee. Java Bean is an instance used for DI in Spring.

Only Spring has a container.

scanBasePackages = {“com.ascending.training”}: tell spring which package to scan.

@Repository (For DAOs): tell spring which class to scan for injection. This is the dependency part.
@Service (For service layer): is similar to Repository, Spring will also scan this part.
@Component (general classes): similar to the other two @Repository and @Service
For Spring, @Repository @Service and @Component are the same, Spring will scan all of them as dependency resources for DI. The different names are only for human to read.

@Autowired: this is the injection part.

@RunWith(SpringRunner.class): how to run the unit test
@SpringBootTest(classes = AppInitializer.class)

QUESTIONS:
How do you understand REST? (representational state transfer) [This is a piece of jargon.]

@JsonIgnore: sometimes we need it, but sometimes we don't.

Week 5, Random 3 (important)

Question 1: How to solve the issue that one result returned multiple times on the front end:
1. query.list().stream().distinct().collect(Collectors.toList());
2. Use Set: Set set = new HashSet(list); List list = new ArrayList(set);
3. Hibernate: change query.list() to
   query.setResultTransformer(Criteria.DISTINCT_ROOT_ENTITY).list(); (this is deprecated)

Question 2: Sometimes, we need @JsonIgnore, but sometimes we don't need it. To this purpose, we can change the mode class:

```
public Set<Account> getAccounts() {
    try{int size = accounts.size();} //no particular purpose, just try to use accounts
    //When accounts is accessed, Hibernate will try to get the data (LAZY).
    //But at this time, session is closed, so there will be an exception.
    //As a result:
    //if accounts doesn't exit (not fetched by HQL), it throws an exception. Then in catch, it returns null;
    //if accounts does exit (when join fetch is used), nothing is changed. It will return accounts at the end.
    catch (Exception e) {
        return null;
    }
    return accounts;
}
```

QUESTIONS:
@RequestBody, how to get two different-type objects? 【use wrapper class with those two objects】

```
public class Wrapper {

    private Department department;
    private List<Student> students;

    public Department getDepartment() {
        return department;
    }

    public void setDepartment(Department department) {
        this.department = department;
    }

    public List<Student> getStudents() {
        return students;
    }
```

```
    public void setStudent(List<Student> students) {
        this.students = students;
    }
}
```

JSON input:

```
{
        "department":{
                "name":"AOES",
                "description":"atmospheric, oceanic, and Earth Sciences",
                "location":"Research Hall"
            },
        "students":[{
            "stName":"lyu1",
            "firstName": "Liang1",
            "lastName": "Yu1",
            "email": "@gmail",
            "address": "Fairfax"
                    },
                    {
            "stName":"lyu2",
            "firstName": "Liang1",
            "lastName": "Yu1",
            "email": "@gmail",
            "address": "Fairfax"
                    }]
}
```

Week 5, Random 4 (Postman, Bash)

postman: collection, and environment variable.

Bash: change permission.

QUESTIONS:

For account withdraw function, add in controller or service layer? 【service layer】

Which part is dispatcher servlet and handler mapping? 【cannot see it in Spring frame work.】

How to control filter direction? for incoming data or outgoing data?【code before doFilterChain is for coming request; code after doFilterChain is for outgoing data.】

Week 6, Class 1 (Authentication, Authorization, ManyToMany)

authentication: who you are
authorization: what you are allowed to do.

stateful (session based): Once log in successfully, session is created on the backend, server sends a session ID to client (each time logging in the ID is different).

stateless (token based): Authentication and authorization can be on separate server. User logs in, then server creates and sends token to user. Token has the user authorization info, so next time when user makes some operations, server only checks the token then decides whether let the request pass or not.

Java servlet filter: pre-processing and post-processing of data sending between user and server.

DispatcherType.*REQUEST (only for data coming in not for data sending to users)*

urlPatterns = {"/*"} (which url will be filtered.)

JWT: JSON web token.

JWT format: header.payload.signature (separated by dot)

header:{alg:"HS256",typ:"JWT"} (alg: encode method)
payload (or claims, define by yourself): jti (id), iat (issue at), iss (issued), exp (expire)
signature:
1. encodedString = base64UrlEncode(header) + "." + base64UrlEncode(payload)
2. hashValue=hash(encodedString, secret-key)
3. Signature = base64UrlEncode(hashValue)

why we need encode? (no conflict with the default keyword of the url)

Week 6 Class 2 (cont)

regular expression: "." any string, "$" last char
"^(.*?) ": every thing before the first space (there is a space between "")

Three mistakes:
1. add url password in the VM option
2. encode in the getPassword() after getting info from postman, the encoded data is the value saved in the java object.
3. secret.key cannot be too short or simple

Try:
command out setter in model class so we can test whether json format data can be saved into the object.

Filter order: we need configuration file "web.xml" or Spring will run the filter ordered by the filterName

Week 6 Random 2 (encode and decode)

I was late.
encryption/decryption
need a third party certificate: certificate authority (CA).

Week 6 Random 3 (Filter)

Why do we need RESTful API: like standard port. so convenient to use for different app.
@Requestbody (Payload)

In different filter: we can do authentication, in another filter we do authorization, logging.

HOMEWORK:
check: a request goes to filter first or goes to controller first?

check: request path order?

check: no setter, can we still set the object from the json input?

check: control the filter direction before and after the chain.doFilter.

how to run an html file in intellij?

how to run the example code using Spring?

Week 7, class 1 (Algorithm: data structure)

stack: first come, last go.
【double stack】

queue: first come, first serve. LinkedList()

list:
when define list, we can jump index. 【does not work. This is wrong.】
try to print no value (null or 0?)
what is the difference between arraylist and list?

map: key must be unique, value can be the same. 【not true】
hashcode() and equals()
what's the difference between hashmap and map?
hot key bucket. (a lot of values in one bucket)
same key value in different bucket, is that possible? 【Yes】

time complexity: O(n).

Binary tree (DFS & BSF): deep first and broad first
traversal,
class Node{
int val;
Node left;
Node right;
}

Queue.size() is the stop criteria.

Week 7 Class 2 (cont, 2D array)

Linked List: 链表，one points to a next one.
skills (double pointers): using fast and slow pointers to find out the half point.
find the merge point.

Array: find the same value in list. create another list with the same length. The value in the first list becomes the index in the second list, then count the number as the value in the second list.

Fibonacci: use space to save time, so we save some value. time complexity O(2^n); 【try】

backTracking: search online.

Week 7 Random 1 (servlet)

if web app is a hotel, servlet is like a front desk of a hotel.

web app three important components: listener, filter, servlet, controller.

doGet vs doPost:
doGet has limited data size; doPost has unlimited data size.
doGet need to use URL, not good for password
doGet has bookmark function.

Forward vs Redirect:
1. forward happens at server side and the same server; redirect happens at client side.
2. request is forward inside the same server; redirect is giving the request another URL to visit, could be another server.

Single sign on: log in once, use different app, instead of logging in for each app.

nowadays front end:
front end: react JS, angular JS,
another front end: JSF framework, html, (code is clean)

old times, front end: JSP and ASP, these two method the code is messy. Web content and content fond are mixed.

try filter order in Spring. 【true, Spring call filter in the filter class name alphabetical order】

Week 7, Random 2 (listener)

web server>web container (servlet container)>web application (RESTful API)
tomcat is a web server and a web container

Web container creates events and listener captures the events so we can do something.

listener type: life cycle changes and attribute changes.
Application level: ServletContextListener and ServletContextAttributeListener (attribute (key/value): add, delete, replace)
Session level (stateful server): HttpSessionListener and HttpSessionAttributeListener
Request level (not common) : HttpRequestListener and HttpRequestAttributeListener.

ServletContext is an object, which will be created by the web servlet. Web container will copy the information in the web.xml to the ServletContext. When Servlet opens and closes, there will be an event and ServletContextListener will capture this event so we can do something. Example, add List<> country.

Week 8 Class 1 (AWS integration, S3)

S3: simple storage service, (1) cost, (2) ease to access.
traditional server: room+power+air conditioning+network+server+human resource. That's why it is expensive.

concepts: buckets (like a container), objects (data and metadata), key (like file location and file name, we can identify the object), prefix (like folder directory),regions (IDC1, IDC2, IDC3, IDC4), pre-signed URL (other people can visit your source)

@Bean: method level, for the third party code.
@Repository, @Service, @Conponent: class level

@bean
@scope:
1. singleton (only create instance or object once, every one uses the same object)
2. prototype (create different instance for different classes, i.e. logger)

3. request (when request received, object will be created, when request finished, object will be destroyed)
4. session
5. globe session

S3 bucket allows files with the same file name, as long as the versioning is open.

MultipartFile mean a single file contains different parts.

Week 8 Class 2 (SQS)

SQS: simple queue service.

We need to save message such as logs, email messages, which might have a very large size. We also need to handle some urgent request first, leaving the un-urgent services, such as sending the emails, handled when not busy.

HOMEWORK:

do a unit test for File service

Authentication and authorization (security code): write it yourself.

try logger in different class, do I have class name?

try listener.

try consume the message in SQS, and send it to client. (when user uploads a file, save it to S3, send a message to SQS, and then consume this message and send a URL link to user's email (send grid), when user clicks this URL, he/she can download the file)

Week 8 Random 1 (AWS)

Some AWS services: EC2 (computing), RDS, S3, SQS, IAM (secret key), kinesis (grab data online), cognito (JWT), SES (not as good as send grid)

DefaultAWSCredentialsProviderChain() will get the key and secret and log in S3 chain order:
1. environment variable (export varaiblename='blabla')
2. java system properties (VM option)
3. default profile (~/.aws/credentials)
4. ECS
5. instance profile credentials.

Week 8 Random 2 (Garbage Collection)

https://blog.csdn.net/justloveyou_/article/details/71216049

JVM architecture: Variable is stored in stack. Data is stored in Heap.

What: An automatic process of JVM to inspect heap memory. Identify which objects are in use, which are not. Delete those unused objects. When garbage collector runs, every thing stops.

Why: prevent memory leak.

Heap: young generation (YG) and old generation (OG).
YG: eden (newly created objects) and survivor (S0, S1)
OG:

Steps: 1 marking 2 sweeping (delete) 3 compacting

minor garbage collector takes care of the YG region;
major (full) garbage collector is in charge of the OG region.

QUESTIONS:
Why do we need Path path = Path.get("/user/temp.txt")? Why don't use String path = "/user/temp.txt"? 【1. Convenient, Path class has some methods for path operations. 2. The following method needs Path as input】

```
InputStream reader = new BufferedInputStream(
  object.getObjectContent());
File file = new File("localFilename");
OutputStream writer = new BufferedOutputStream(new FileOutputStream(file));

int read = -1;  【-1 represents end of file】

while ( ( read = reader.read() ) != -1 ) {
   writer.write(read);  【write to the buffer byte by byte, when the buffer is full, do IO operation】
}

writer.flush();  【write the remaining part in the buffer to disk】
writer.close();  【close writer buffer】
reader.close();  【close reader buffer】
```

How to get more messages from SQS? I can only get two messages after setting
receiveMessageRequest.setMaxNumberOfMessages(10);
receiveMessageRequest.setWaitTimeSeconds(20);

How do we use standard SQS queue if it is not FIFO? I don't even know what is the message I can get from the queue.

Web listener does work for Spring frame work? It does work for tomcat. 【the embedded Tomcat in Spring is different from the real Tomcat.】

In security filter uses "@Autowired Logger logger;" still correct. Why do we change this logger in class? 【Because I did not pack the modified war file, it still runs the old war】

Week 9 Class 1 (software testing)

https://www.vogella.com/tutorials/Mockito/article.html#mock-object-generation
Basic categories of testing:

1. White box test: unit test is white box test. When you test it, you know the inner part of the code.
2. Black box test: You don't know the code when you test it.
3. Grey Box test (not common): The tester knows both the inner code structure and the quality of the project.

Level of testing:
1. Unit test: test an individual component or method.
2. integration test: test the integrated components. higher than unit test
3. System test: test the entire system

Testing Doubles: replacement for the real dependencies. (because we don't want the impact of other components that we don't want to test)

testing doubles includes:
1. Dummies: objects passed around but never used.
2. fakes: a simplified version of the real third party function.
3. stubs: provide canned answers to calls made during the test. (give a known input value.)
4. spies: are objects that are record wrapped objects interactions with other objects.
5. mocks: are objects that simulate the real objects.

verification:
state verification: check the result
behavior verification: check the process, whether a certain object is called.

test framework:
JUnit, mockito, powermock, easymock

KEY POINTS:
https://www.vogella.com/tutorials/Mockito/article.html#mock-object-generation

Limitations:
1. mockito cannot mock final class. So you need to add a file.
2. cannot mock a static method and private method.

TRY:
try to run mockito in project not only in test.

test overide and hide.

private constructer. review modifier.

Week 9 Class 2 (multithreads, Tomcat)

Similar to tomcat: wildfly/JBoss.

To use Tomcat, change the following three:
1. `public class AppInitializer extends SpringBootServletInitializer`

2.
```
public class SecurityFilter implements Filter {
    private Logger logger = LoggerFactory.getLogger(this.getClass());
```

3. in pom.xml <packaging>war</packaging> //so mvn will pack a mvn file.

4. Then run to pack current project into a war file:
mvn -Dmaven.test.skip=true clean compile package

5. Go to /usr/local/Cellar/tomcat/9.0.24/libexec, run "./bin/catalina.sh run"

deploy (部署):
copy your war file to /usr/local/Cellar/tomcat/9.0.24/libexec/webapps/ROOT.war

(Why is the name is ROOT.war? Tomcat will deploy the application under the root directory in the local server)

(Tomcat steps: 1. explode the war file; 2. get two folders: META-INFO, WEB-INFO)

Then run Tomcat under bin folder: ./catalina.sh run

To get the VM option values, create setenv.sh file (set it runnable) in Tomcat bin folder. add your env variable:

```
export JAVA_OPTS="$JAVA_OPTS -Ddatabase.driver=org.postgresql.Driver"
export JAVA_OPTS="$JAVA_OPTS -Ddatabase.dialect=org.hibernate.dialect.PostgreSQL9Dialect"
export JAVA_OPTS="$JAVA_OPTS -Ddatabase.url=jdbc:postgresql://localhost:5431/mypostgresDB"
export JAVA_OPTS="$JAVA_OPTS -Ddatabase.user=admin"
export JAVA_OPTS="$JAVA_OPTS -Ddatabase.password=123"
export JAVA_OPTS="$JAVA_OPTS -Dlogging.level.org.springframework=INFO"
export JAVA_OPTS="$JAVA_OPTS -Dlogging.level.com.ascending=INFO"
export JAVA_OPTS="$JAVA_OPTS -Dsecret.key=23414asfdhp*^%"
export JAVA_OPTS="$JAVA_OPTS -Daws.accessKeyId=AKIAJAKFI6XUNPZHCSJQ"
export JAVA_OPTS="$JAVA_OPTS -Daws.secretKey=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"
```

Week 9 Random 1 (mock test)

difference between stub and mock?
mock always returns a null value. After mocking, we need to use stub to return a known value.
why stub can return a value?

```
AmazonS3 amazons3 = Mockito.mock(AmazonS3.class);
S3Object obj = Mockito.mock(S3Object.class);

when(amazons3.get("car-bucket","test.txt").thenReturn("hello");
when(amazons3.get(anyString(),anyString).thenReturn(obj);

Mockito.verify(S3, times(1)).getObject("car-bucket", "test.txt");
```

Spring Framework, @Profile

HOMEWORK:
use mockito for test

rewrite security filter

Week 9 Random 2 (Multithreading)

Process: an executing instance of application.
Thread: exist within process. It is path of execution within the application.

Difference:
1. process has its own addresses space. Thread shares the space of process.
2. the messaging between threads is easer.
3. thread is light-weighted.

Thread model:
1. cooperative: run thread one by one, until one is done or yield or blocked. (Windows 95 or earlier)
2. preemptive: The system will control the maximum time one can use for one job.

Java priority level: 1-10, 5 by default. Java uses preemptive mechanism for different levels. Time-sliced for the same level.

Two types of threads in Java:
1. User: keep running even main method stops.（用户线程）
2. Daemon: stop when main method stops. (守护线程)

Runnable and Callable:
1. callable has return type
2. callable can throw exception
3. One overrides run, one overrides call

Two methods to run a thread in ExecutorService class:
1. submit: Thread, runnable, callable; It has a return value, i.e. Future.
2. execute: Thread, runnable only

TRY:
create Daemon thread

Week 9 Random (Multithreading cont)

1. Synchronized : both method and code block. The synchronized park should as small as possible.
2. Lock framework:
3. volatile: only for variable, don't catch the shared variable.

Week 10 Class1 (Mockito, https in Tomcat)

doReturn.when vs when.thenReturn:
1. is the same for mocked objects
2. different for spied objects: when.thenReturn will call real method on spied object. but doReturn.when will not call the real spied object (not side effect).

How to use https in Tomcat (under homedir of Tomcat):
1. keytool -genkey -alias liang -storetype PKCS12 -keyalg RSA -keystore ./conf/tomcat-keystore.jks
2. enter keystore password: 123456
3. answer some questions.
4. show the key: keytool -list -keystore ./conf/tomcat-keystore.jks

5. add in the server.xml file:

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
    maxThreads="150" scheme="https" secure="true"
    clientAuth="false" sslProtocol="TLS"
    keystoreFile="conf/tomcat-keystore.jks"
    keystorePass="123456" />
```

6. log in: https://localhost:8443/manager
7. change 8443 to 443 (443 is the default port number, so you don't need to input :8443, you can just input https://localhost/manager).
8. user: admin, password: admin.
9. To use https from Postman, we need to set Preference SSL certificate verification to OFF.

pack and unpack:
tar zcvf/tar zxvf

Week 10 Class 2 (AWS CLI, SQS Listener)

1. sudo easy_install pip (install python)
2. pip install awscli --upgrade --user (install aws command line interface)
3. create ~/.aws, add two files: config and credentials.
4. save AWS key and password in File credentials as follows:

```
[default]
aws_access_key_id = xxx
aws_secret_access_key = xxx
```

HOMEWORK:

1. try aws CLI.
2. security filter.
3. upload multiple files through postman to S3.

QUESTIONS:

https://stackoverflow.com/questions/109383/sort-a-mapkey-value-by-values

```
public class MapUtil {
    public static <K, V extends Comparable<? super V>> Map<K, V> sortByValue(Map<K, V> map) {
        List<Entry<K, V>> list = new ArrayList<>(map.entrySet());
        list.sort(Entry.comparingByValue());
        Map<K, V> result = new LinkedHashMap<>();
        for (Entry<K, V> entry : list) {
            result.put(entry.getKey(), entry.getValue());
        }
        return result;
    }
}
```

Week 10 Random 1 (micro service)

What: each service is a unit function, (one single service per container)
Why: different service is independent to each other. easy to maintain.

Week 10 Random 2 (Multithreading)

Lock vs synchronized:
1. lock is explicity locking; synchronized is implicity locking
2. lock is more flexible to lock and unlock.

Some concepts:
1. deadlock
2. livelock
3. starvation
4. race condition

Week 10 Random 3 (redis, not come)

https://github.com/ascending-llc/spring-redis

How to setup redis in spring boot?
1. in pom.xml add: <dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-data-redis</artifactId>
   </dependency>
2. in application.properties add:
   #redis config
   spring.cache.type=*redis*
   spring.redis.host=localhost
   spring.redis.port=6379
3. in appinitializer add: @EnableCaching
4. add the following in front of the controller you want to use cache:

```
@Cacheable(value = "departments")
@CachePut(value = "departments", key = "#department.id", unless =
“#department.name==null”)
```

5. Model class needs to implement Serializable (object needs to be serializable first so the object can be transferred to cache)

QUESTIONS:

1. what is this for in redis demo?

<!-- for JPA support -->	
	<dependency>
	<groupId>org.springframework.boot</groupId>
	<artifactId>spring-boot-starter-data-jpa</artifactId>
	</dependency>

2. How to set cache aside?

3. Why is there still nothing in redis (running redis-cli keys *) for the second request but I do find 1 hit and 1 miss?

4. Error for write-through request: Request processing failed; nested exception is org.springframework.expression.spel.SpelEvaluationException: EL1007E: Property or field 'brand' cannot be found on null.

5. When I use database I don't need serialization. Why do I need serialization when I use redis?

Week 11 Class 1 (docker build/run image)

Continuous Integration and Continuous Deployment (CI/CD).

Docker: easy to migrate.

Dockerfile and Docker image:
1. both are static
2. Dockerfile is readable, docker image is not readable.
3. docker image is generated based on Dockerfile.

"docker run -it ubuntu /bin/bash" (i: interactive, t:terminal)
"docker build -t my_image:0.1 -f Dockerfile ."(t: tag, build will run the commands in Dockerfiles)
"docker run my_image:0.1 printenv"
"docker inspect CONTAINER_ID"
"docker run -p 80:8080 -d mytomcat:1.0 "

Some Dockerfile commands;
FROM, ENV, COPY, RUN (effect when build), CMD (effect when run), EXPOSE, ARG

QUESTIONS:
1. How to edit a new file in Dockerfile?

RUN echo "Some line to add to a file" >> empty.txt

Week 11 Class 2 (CI/CD)

CI/CD: After developer pushing code to github, CI/CD helps automatically test -> package a war file -> build an image -> deployment to the server.
CI: is from pushing to github to the test part.
CD: is from package to deployment.
CI/CD: online process, automatically. CI/CD are only available after docker and cloud coming out.

DevOps: people in this position helps build and maintain CI/CD.

ECR: docker image storage.
S3: store war file.

Week 11 Random (stateful, stateless)

stateless: authentication is stored in the client side (in the token) and authorization every time. can change server convenience. shortage: every one can log in with the same token.
stateful : authentication once. safer (other people cannot use this sessionID).

HttpServletResponse code 302: found.

QUESTIONS:
1. Does JWT have anything to do with JSON? Why is it called JWT?
2. Does this sentence use base64URLEncode? JwtUtil.java line 61: return builder.compact();
3. Where is header? JwtUtil.java line 58: JwtBuilder builder = Jwts.*builder*().setClaims(claims).signWith(signatureAlgorithm, signingKey);

Week 11, Random (serializable, de-serializable, not there)

Serialization is a mechanism of converting the state of an object into a byte stream. Deserialization is the reverse process where the byte stream is used to recreate the actual Java object in memory. This mechanism is used to persist the object.

We need to implement the serializable interface to make the object serializable. Serializable is a **marker interface.** There is no method in the serializable interface.

After serialization, only the state fields will be stored and only instant variables will be serialized. Serialization doesn't include method information and static variables.

For some information such as password or ssn, you don't want to serialize, you can use transient key word, for example:
private transient int ssn;

SerialVersionUID

writeObject/readObject: Encrypt/DeEncrypt.

Week 12, Review

What, Why and How (2W1H)

1. **1WARMUP**
2. DBMS
3. SQL (DDL, DML, DCL, TCL)
4. Docker (what is definition, image, container, why container? (3 points))
5. Java Basic: four modifiers
6. maven
7. JDBC (java data base connectivity)
8. flyway
9. **2STRETCHING**
10. Java Basic: Encapsulation, Inheritance, Polymorphism.
11. JUnit
12. Logger
13. Git
14. **3HIBERNATE**
15. 2W1H
16. architecture: configuration (2 parts), sessionFactory, session, query, cache
17. annotations: @Entity,@Table,@Column,@Id,@JoinColumn
18. @OneToOne, @OneToMany, @ManyToOne, @ManyToMany
19. SessionFactory (thread safe), Session (not thread safe)

20. inverse side v.s. owning side
21. FetchType.LAZY v.s. FetchType.EAGER
22. join v.s. join fetch?

23. 4SERVICELAYERANDDI

24. Service Layer: 2W1H
25. DI: 2W1H
26. annotations: @SpringBootApplication, @Component, @Service, @Repository

27. 5RESTFUL WEB SERVICE

28. Web Service: 2W1H
29. Restful Web Service: 2W1H
30. five http methods
31. Idempotance
32. Controller: 2W1H
33. @PathVariable, @RequestBody, @RequestHeader, @RequestParam

34. 6JWT

35. What is authentication and authorization
36. Stateful and stateless: 2W1H
37. What is filter?
38. ManyToMany
39. JWT: 2W1H
40. JWT format? (HS256, base64URLEncode)

41. 7AWSINTEGRATION

42. Credential chain
43. S3: 2W1H
44. bucket, object, key, prefix, region
45. SQS: 2W1H
46. JMS messaging model: point-to-point, publish-subscribe.

47. 8MOCKTESTING

48. white box, black box, grey box.
49. test doubles, stub, spy, mock
50. verification: state/behavior
51. What are they: @Mock, @Spy, @InjectMocks
52. What is the difference: when(...).thenReturn(...) v.s. doReturn(...).when(...)

JAVA BASIC INTERVIEW QUESTIONS

1. See another file InterviewQuestions.pdf
2. lambda expressions
3. Stream

HOMEWORK:

Try to use JUnit or other APIs without maven.

Try to test controller by unit test, not using Postman.

Try @InjeckMocks (https://tedvinke.wordpress.com/2014/02/13/mockito-why-you-should-not-use-injectmocks-annotation-to-autowire-fields/)

Try to use Hibernate configuration xml file. (hibernate.cfg.xml, hibernate.properties)

Try hibernate no student in department mapping class (in student model class, department is still there)

Try to combine frontend with the backend.

Try CI/CD.