лк лшж ! ª л ª щэк жшщы л ª л ж ыª ª ª л ª з шж ª жыл ! ª л э ˘ шлн к л л щ˘ л ª
ª з шж ª ж

Университет ИТМО»

Факультет ПИиКТ

Дисциплина: Технологии веб-сервисов

Лабораторная работа 1

ª эш л ж ª ˘ ж ˘ ж щ ж щл жщ ! л ж

шл ª к ж жыл ! лш ж˘ л к шл Ф ж ª ˘

шэ ж

ж ы) лылшв эш

Задача

кж а шжз а ыл ьшлз эльщ щ а кжы! ьжз э зжлкж щ а клшн ж э л л лл ы а л жьж н л шлж а жы! а а н а щы! а щж а з а з ж а л щ а а ! В)щпш щк ж лк а щж к а н лшлк ж жы! щ ль а к щпш щк ж˘лщы л жш э л ы а

лз)щпш щ л а з а к а шлж а жы! кл b b ff) ш а н л) ш а н л ш шлж ж кл) ш а н л щ лк эл ы ж щы а ш а л щпш лшж ш а н л жщы ш а ы! щы а ˘ кж а щэ лщы ы! л а л а кл щпш щк

кл а щ ы шж шжз а ы шж шжз а ыж щпш щ а щ лк эл ыьж н л шж шжз а ьжы! л ьщ а л а щ а ! а л ш а н л л

Выполнение

л а ы а ш ꞩt i ȷ ȷ ȷ ȷ) ff) ff ȷ ff) ff i ȷff

ш шлж ж з а л щ лкэ л ˘ ж

а кж шл а ы а ш
а кж ˘ жз а ш а л ьжщ щ а ! а ж л b ff
ш а л ыз шжкл л ж а кэ
b b) ff ff щшлж ж л щпш лш а ˘ жщы ш а н л
b) ȷff щшлж ж л а щ а ! а а л ьж
b)ꞩt а b)bꞩg шлк щьж щпз ьлшя л щ
ж а кл щы щbꞩg жщщ л лш шэ ьщ ж а щ а л
я ж а а я эшж щпш лшж
жщы ш а л ш а лщщж ы а жы ˘ лщ а а ж ж ж а кжщ а а ! ff b ff
ꞩt i ȷ ȷ ȷ ȷ) ff) ff ȷ ff)
ff i ȷff b ff ff b ff а
жщы ш а л ш а лщщж ы а жы ˘ лщ а а я а ш жы ш а ж а кжщ а а !
ꞩt ff
а к ˘ л ꞩt ff) ȷ э к а ьш а ж˘лщы ж а кж ш а лщщп
шж шжз а ы ˘ жлы щпз щ лкэ э а я эшж
b а ш жы ш а ж л лшл а щкщы ш а
а ьш а ! ˘ ш а з л а
ш а лш жк а з ж л щ ˘ а з а ! ˘ я ж а шл а ы а ш
ы а жы ˘ лщ а л я а ш жы ш а ж л а кж
ff ы а жы ˘ лщ ж ж а кж
а ш жы ш а ж л я ж а

жшьшªл шª лшщ к шзªш ªз ª л ª лшж лшщ b ff
ªк ªьª шл ж і ј St і ј))
ff) ff ј ff) ff і ff b ј b ff ᵇ
жшьшª ж л лкн л ьж лшщ жльª b ff к
л ьшж ª ж ª ª ª ьшª лшщ шл з з ªыл
ª я эшж ff ff) St ffк экªз ª ª щª ! ª ж
шлш лшж ш ªн л ј ᵇ ш шзªш л шлш лшжк к л ª жшлш лшл
ш ªн л
жшьшª жзж к ж і ff к шж лшы ж щª ª ! ff)
St ff
жшьшª ж л b b ff ј ᵇщ ª ª ! ª ж к b ff
лж ж шлш лшª ˇжшы щ щª ! ª ж л В) яэ л
ª щ ж ª шэ ªшы
лж ж ж ьª жы ˇлщª л лшж ªкжк кªшьэ ж
шлш лшж жªщª л
лж ж ª щª ! ª ª л ьж

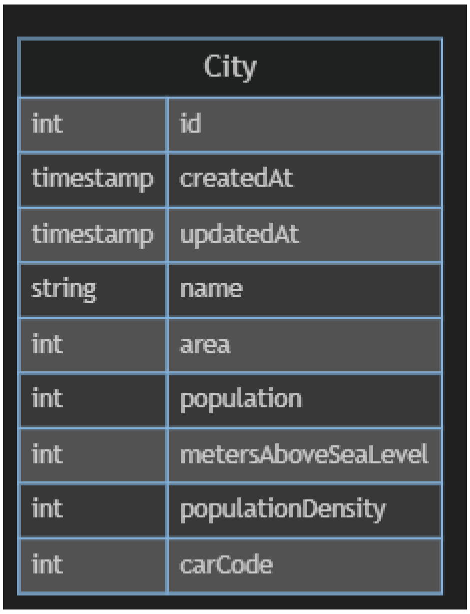
Реализация серверной части

ж ˇж жз ª шлкл л шьил

- b b
- b b b В) щ л я ж к шлж ж лз шлш шжщ щª ! ª ж л В шªьª ª ж
- лж ж щ л я ж В) ªы ª ж ª
- О і ж ˇлшы л шлш шжª шª ж
- ј ff b ffк шнзªы шзжª к ж щª ! э
- ј ff b ff b ј b к щª ! ª ж b b b b ј b ј В
- ᵇ bᵇ к шлж ж шж з ж к ж жьª жы ˇлщª ª
ш л л ш шьжшыл ш ªн л
- b ff к шлж ж жшщ жжш э л ьª ª щª ! ª ª
ш ªн л
- ff к ªк ˇл ffSff ff ᵇ nff ј щ э ˇжл b b ff
л ьжьж ж щª ! эльщ b b
- О к экªзшы жшнзªы щ льыылшж шлыылшж ª шьшэ ьª шж

ª шьж л л щэ ª шы

жщª шьж л жщэ ª шь! ª шª к ж ˘ж ж щ лк э л ª



City	
int	id
timestamp	createdAt
timestamp	updatedAt
string	name
int	area
int	population
int	metersAboveSeaLevel
int	populationDensity
int	carCode

ª к ж л ьлшя л щкщлш щк

ﬆ i ˩ ˩˩ ˩) ﬀ) ﬀ ˩ ﬀ)

ﬀ i ˩ﬀ b ﬀ b b)

ﬀ ﬀ b˩ ᵲb b i ˩˩ ﬀ ˩ ﬀ ˩ ﬀ b ﬀ ˩ ᵇ ﬀ ˩ ﬀᵲb b

лш щ ª ª лы ш л ы! щ щª я ! ышª ª шлк л л ª ª я ª ш жьж э ж ж к жн к ª ª л л ьжя ! ышж ª л ж˘л л ª ª ыª шª э з эк лы шª щ ª к ы! я ! ышж

```
package guldilin.service.interfaces;

import guldilin.dto.CityDTO;
import guldilin.dto.FilterArgumentDTO;
import guldilin.dto.PaginationDTO;
import guldilin.dto.PaginationRequestDTO;
import guldilin.exceptions.FieldIsNotFilterable;
import jakarta.jws.WebMethod;
import jakarta.jws.WebParam;
import jakarta.jws.WebService;
import jakarta.validation.Valid;
import java.util.List;

/**
 * CityService interface.
 */
@WebService(
        name = "CityWs",
        serviceName = "CityService",
        targetNamespace = "http://service.guldilin",
        portName = "CityPort",
        wsdlLocation = "META-INF/wsdl/CityService.wsdl")
public interface CityService {
    /**
```

```
     * Find elements by field-value filters.
     *
     * @param filters List of field-value filters
     * @param pagination pagination information
     * @return Found elements
     * @throws FieldIsNotFilterable for incorrect filters argument
     */
    @WebMethod
    PaginationDTO<CityDTO> findByFilter(
            @WebParam(name = "filters") List<FilterArgumentDTO> filters,
            @WebParam(name = "pagination") @Valid PaginationRequestDTO pagination)
            throws FieldIsNotFilterable;
}
```

ª к ж л ьлшя л щжшл ª ьª ш

шжз ª ы щ з ª шл ˘ л ª щ ª ! ª жы! ȷ ff ȷb bsȷ ьж ж щл ª
ª ª ! ж з ª лл эк ª з ª щьш ª ы! ж ш ª щ щ ш ª ª ! ª ˘ лщы ª
ш л л я !ьш ª

```
package guldilin.repository.interfaces;

import guldilin.dto.FilterArgumentDTO;
import guldilin.dto.PaginationRequestDTO;
import guldilin.entity.Mappable;
import guldilin.exceptions.FieldIsNotFilterable;
import jakarta.persistence.EntityManager;
import jakarta.persistence.criteria.CriteriaQuery;
import java.util.List;
import java.util.Optional;
import org.hibernate.Session;

/**
 * Generic Interface for Entity Repository.
 *
 * @param <T> Entity class
 */
public interface CrudRepository<T extends Mappable> {
    /**
     * Find all Elements by CriteriaQuery.
     *
     * @param criteriaQuery CriteriaQuery
     * @param pagination information about pagination properties
     * @return Result list
     */
    List<T> findByCriteria(CriteriaQuery<T> criteriaQuery, PaginationRequestDTO
pagination);

    /**
     * Count elements in database by criteria.
     *
     * @param criteriaQuery CriteriaQuery
     * @return Number of elements
     */
    Long countByCriteria(CriteriaQuery<Long> criteriaQuery);

    /**
     * Find item by id.
     *
```

```
     * @param id Item id
     * @return Optional Entity
     */
    Optional<T> findById(Integer id);

    /**
     * Creates Entity Manager.
     *
     * @return EntityManager
     */
    EntityManager createEntityManager();

    /**
     * Create CriteriaQuery by filter arguments.
     *
     * @param filterArguments list of filter fields arguments
     * @return Generated CriteriaQuery
     * @throws FieldIsNotFilterable if some filter fields are incorrect
     */
    CriteriaQuery<T> createFilterQuery(List<FilterArgumentDTO> filterArguments)
throws FieldIsNotFilterable;

    /**
     * Create CriteriaQuery for count elements.
     *
     * @param filterArguments List of filters
     * @return CriteriaQuery
     * @throws FieldIsNotFilterable if some filter fields are incorrect
     */
    CriteriaQuery<Long> createCounterQuery(List<FilterArgumentDTO> filterArguments)
throws FieldIsNotFilterable;

    /**
     * Opens Session.
     *
     * @return Session
     */
    Session openSession();
}
```

ж лл ª ы ьª ª ff ff ј ьилшя л щк жщ лкэ льщ ьилшя л щк шф ª шы ј b

лж ж щª к ж ьилшя л щª жкж ª ьж л з ж ж щк ж

ª жшжв ª ы щзк я !ьшж шпш щп

лж ж шж к ж ª э ьил з ж щк щ ш ы

щ ª я эш шª ж шпш щ b bb к ж эщ ж шж

```
package guldilin.config;

import jakarta.annotation.PostConstruct;
import jakarta.ejb.Singleton;
import jakarta.ejb.Startup;
import jakarta.ejb.TransactionManagement;
import jakarta.ejb.TransactionManagementType;
import jakarta.inject.Inject;
import javax.sql.DataSource;
```

```
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
import org.flywaydb.core.Flyway;

/**
 * Implements database migrations running at server start.
 */
@Singleton
@Startup
@TransactionManagement(value = TransactionManagementType.BEAN)
public class FlywayMigrator {
    /**
     * Just logger.
     */
    private static final Logger LOGGER =
LogManager.getLogger(FlywayMigrator.class);

    /**
     * Injected data source.
     */
    @Inject
    private DataSource dataSource;

    /**
     * Run database migrations.
     */
    public void doMigration() {
        LOGGER.info("Start migrations running");
        final Flyway flyway = Flyway.configure().dataSource(dataSource).load();
        flyway.migrate();
        LOGGER.info("Migrations finished");
    }

    /**
     * Startup.
     */
    @PostConstruct
    public void init() {
        LOGGER.info("init FlywayMigrator");
        this.doMigration();
    }
}
```

лж ж ª я эшж b b ffк ªк ˇл

```
/**
     * Produces DataSource.
     * Try to lookup in JNDI first. If not found use env or properties.
     *
     * @param configuration Hibernate Configuration (Injected)
     * @return DataSource
     */
    @Produces
    @Singleton
    public DataSource provideDataSource(final Configuration configuration) {
        LOGGER.info("provideDataSource");
        if (this.datasourceMode.equals(DatasourceMode.JNDI_LOOKUP)) {
```

```
            return this.jndiDataSource;
        }
        PGSimpleDataSource dataSource = new PGSimpleDataSource();
        dataSource.setUrl(configuration.getProperty(AvailableSettings.URL));
        dataSource.setUser(configuration.getProperty(AvailableSettings.USER));
        dataSource.setPassword(configuration.getProperty(AvailableSettings.PASS));
        return dataSource;
    }
```

```
/**
     * Lookup for DataSource in JNDI by Lookup path from properties or env.
     *
     * @return Found DataSource
     * @throws Exception if datasource not found.
     */
    private DataSource lookupDataSource() throws Exception {
        InitialContext cxt = Optional.of(new InitialContext())
                .orElseThrow(() -> new Exception("Missing Application Context to
search datasource"));
        String lookupPath =
Optional.ofNullable(PropertyKey.DB_JNDI_NAME.lookupValue())
                .orElseThrow(() -> new IllegalArgumentException(
                        String.format(ErrorMessages.MISSING_REQUIRED_ENV,
PropertyKey.DB_JNDI_NAME.name())));
        LOGGER.info("Lookup JNDI name: " + lookupPath);
        return Optional.ofNullable((DataSource) cxt.lookup(lookupPath))
                .orElseThrow(() -> new Exception("Datasource lookup failed"));
    }
```

ª шлˇл ª ª ы! ª щ ª жкж ª э ж ж ªн ª
лшлкжы! ˇлшл ff ᵇ ff ſt ſtff ᵇ щэˇжлщ шыªˇ л ж к л
ыª ª лыщ ª ы ж ªк ˇл жªщª лжшэ л ыª ªк ˇл
ªыªш лыжнл ª эыз ы! лшлкж ˇлшл ª шɐнл л ſt ſtff ᵇ

лж ж b b ff ªкэ

жкж ª ыжлщж щªн з ª жэщы! ff bȷ ff к ш
жэщл b b ffrb жш ж шжкл ы! шɐªшэ b rb жэшª л b ff
щ ˇ шж л жɐªш ж щ ªшыл

лж ж ſt ᵃᵇ bſʏ

ж ªкэ ! шлкшыж лы шпз жщщ к кªшɐɵ ж ˇлшл
шªыª ª В л л ш шɐльщ Оя ж ª ªыªш лыжнлщ кжыщ
жщª шлж ж ылшялщ ª ªклщпш лшж жэшª лщпш лшж
ªк ˇл b ff ж л лшж я ж ª ж жшыªшª л ſt ᵃᵇ)bſʏ
ªкэ ªзшнɐªыжяж ª щ ª ª ! ȷ ſt

лж ж ª щª ! ª ª л ыж

л ыª з шнз жы жлы ªк л жш э л ы ª ж к ª щьшª жлы
лыª к st ªb bsŋ

щ ª ª ªк э ! ª л ª ж к

```
/**
 * Executor for main method.
 */
public class MainExecutor implements Executor {
    /**
     * Get executor by passed command.
     *
     * @param args Parsed arguments
     * @return Executor object
     */
    private Executor getExecutorByCommand(final Args args) {
        return switch (args.getCommand()) {
            case find -> new FindExecutor();
            default -> throw new ParameterException("Executor for command not
found");
        };
    }

    /**
     * Build JCommander arguments parser.
     *
     * @return JCommander object
     */
    @Override
    public JCommander buildCommander() {
        return JCommander.newBuilder().addObject(new MainArgs()).build();
    }

    /**
     * Parse main arguments.
     *
     * @param argv Unparsed CLI args
     * @return Parsed arguments
     */
    @Override
    public MainArgs parseArgs(final String[] argv) {
        MainArgs args = new MainArgs();
        JCommander.newBuilder()
                .programName("lab1-client.jar")
                .addObject(args)
                .build()
                .parse(argv);
        return args;
    }

    /**
     * Execute main command.
     *
     * @param argv Unparsed CLI args
     * @throws Exception if execution failed
     */
    @Override
    public void execute(final String[] argv) throws Exception {
        HelpArgs argsHelp = new HelpArgs();
        JCommander.newBuilder().addObject(argsHelp).build().parse(argv);
        if (argsHelp.getHelp() && argsHelp.getCommand() == null) {
```

```
            buildCommander().usage();
            return;
        }
        MainArgs args = parseArgs(argv);
        Executor executor = getExecutorByCommand(args);
        executor.execute(argv);
    }
}
```

```
Usage: lab1-client.jar [options]
  Options:
  * -c, -command
      Command name
      Possible Values: [find]
    -filter, -f
      Filters for read command in format field:operation:value (area:=:1)
      Default: []
    -help, -h
      Show help message (also can be used with -c argument to show help
      message for command)
    -limit
      Results limit
    -offset
      Results offset
  * -url
      Server URL base (example: http://localhost:8080)
```

Вывод