My designed and implemented RESTful service

Yauhen Bichel

April 2022

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem;

import java.util.HashSet;
import java.util.Set;
import javax.ws.rs.ApplicationPath;
import javax.ws.rs.core.Application;
import org.jboss.logging.Logger;


/**
 * Entry Point of the application
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationPath("/api")
public class HolidayApplication extends Application {

        private static final Logger logger = Logger.getLogger(
           HolidayApplication.class);

        private Set<Object> singletons = new HashSet<>();
    private Set<Class<?>> classes = new HashSet<>();

        public HolidayApplication() {
                logger.debug("HolidayApplication");
        }
```

```
    public Set<Class<?>> getClasses() {
        return this.classes;
    }

    public Set<Object> getSingletons() {
        return this.singletons;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem;

/**
 * Constants
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public final class Constants {
        public static final String JMS_PREFIX = "java:/jms/";
        public static final String MESSAGE_QUEUE = "HolidayRequestQueue";
        public static final String JMS_MESSAGE_QUEUE = JMS_PREFIX +
            MESSAGE_QUEUE;
        public static final String HOLIDAY_REQUEST_TOPIC = "
            HolidayRequestTopic";
        public static final String JMS_HOLIDAY_REQUEST_TOPIC = JMS_PREFIX +
            HOLIDAY_REQUEST_TOPIC;
        public static final String JNDI_CONNECTION_FACTORY = "java:/
            ConnectionFactory";

        public static final String QUEUE_KEY_MESSAGE = "json-request";

        public static final String DATASOURCE_LOOKUP_KEY = "java:/PostgresDS";

        public static final String UNCAUGHT_EXCEPTION_MESSAGE = "We are sorry.
             We are working to fix the issue. Please try later again.";

        public static final Integer PERCENT_MEMBERS_ON_DUTY = 60;
```

```
        public static final Integer PERCENT_MEMBERS_AUGUST_ON_DUTY = 40;
}
```

] package com.holidaysystem;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem;

import java.util.HashSet;
import java.util.Set;
import javax.ws.rs.ApplicationPath;
import javax.ws.rs.core.Application;
import org.jboss.logging.Logger;


/**
 * Entry Point of the application
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationPath("/api")
public class HolidayApplication extends Application {

        private static final Logger logger = Logger.getLogger(
            HolidayApplication.class);

        private Set<Object> singletons = new HashSet<>();
    private Set<Class<?>> classes = new HashSet<>();

        public HolidayApplication() {
                logger.debug("HolidayApplication");
        }

    public Set<Class<?>> getClasses() {
        return this.classes;
    }

    public Set<Object> getSingletons() {
        return this.singletons;
```

```
    }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem;

/**
 * Constants
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public final class Constants {
        public static final String JMS_PREFIX = "java:/jms/";
        public static final String MESSAGE_QUEUE = "HolidayRequestQueue";
        public static final String JMS_MESSAGE_QUEUE = JMS_PREFIX +
            MESSAGE_QUEUE;
        public static final String HOLIDAY_REQUEST_TOPIC = "
            HolidayRequestTopic";
        public static final String JMS_HOLIDAY_REQUEST_TOPIC = JMS_PREFIX +
            HOLIDAY_REQUEST_TOPIC;
        public static final String JNDI_CONNECTION_FACTORY = "java:/
            ConnectionFactory";

        public static final String QUEUE_KEY_MESSAGE = "json-request";

        public static final String DATASOURCE_LOOKUP_KEY = "java:/PostgresDS";

        public static final String UNCAUGHT_EXCEPTION_MESSAGE = "We are sorry.
             We are working to fix the issue. Please try later again.";

        public static final Integer PERCENT_MEMBERS_ON_DUTY = 60;
        public static final Integer PERCENT_MEMBERS_AUGUST_ON_DUTY = 40;
}



/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
```

```
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.common.exception;

import javax.ws.rs.core.Response;
import javax.ws.rs.ext.ExceptionMapper;
import javax.ws.rs.ext.Provider;
import org.jboss.logging.Logger;

/**
 * Handle exception and return user friendly message about error in server
    side
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Provider
public class UncaughtExceptionMapper implements ExceptionMapper<Exception> {

    private static final Logger logger = Logger.getLogger(
        UncaughtExceptionMapper.class);

    @Override
    public Response toResponse(Exception ex) {
        logger.error(ex.getMessage(), ex);

        return Response
                        .status(Response.Status.INTERNAL_SERVER_ERROR)
                        .entity(ex.getMessage())
                        .type("text/plain")
                        .build();
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
```

```
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.holidaysystem.common.exception;

import org.jboss.logging.Logger;

/**
 * Raises when the system contains the same records
 * @author yauhenbichel
 *
 */
public class RecordDuplicateException extends RuntimeException {

        private static final Logger logger = Logger.getLogger(
            RecordNotFoundException.class);

    public RecordDuplicateException(String errorMessage) {
        super(errorMessage);
        logger.error(errorMessage);
    }
    public RecordDuplicateException(String errorMessage, Throwable err) {
        super(errorMessage, err);
        logger.error(errorMessage, err);
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.holidaysystem.common.exception;

import org.jboss.logging.Logger;
```

```
/**
 * Raises when the system does not contain required record
 * @author yauhenbichel
 *
 */
public class RecordNotFoundException extends RuntimeException {

    private static final Logger logger = Logger.getLogger(
        RecordNotFoundException.class);

    public RecordNotFoundException(String errorMessage) {
        super(errorMessage);
        logger.error(errorMessage);
    }
    public RecordNotFoundException(String errorMessage, Throwable err) {
        super(errorMessage, err);
        logger.error(errorMessage, err);
    }
}
```

] package com.holidaysystem.common.exception;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.common.exception;

import javax.ws.rs.core.Response;
import javax.ws.rs.ext.ExceptionMapper;
import javax.ws.rs.ext.Provider;
import org.jboss.logging.Logger;

/**
 * Handle exception and return user friendly message about error in server
    side
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Provider
public class UncaughtExceptionMapper implements ExceptionMapper<Exception> {
```

```
    private static final Logger logger = Logger.getLogger(
        UncaughtExceptionMapper.class);

    @Override
    public Response toResponse(Exception ex) {
        logger.error(ex.getMessage(), ex);

        return Response
                        .status(Response.Status.INTERNAL_SERVER_ERROR)
                        .entity(ex.getMessage())
                        .type("text/plain")
                        .build();
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.holidaysystem.common.exception;

import org.jboss.logging.Logger;

/**
 * Raises when the system contains the same records
 * @author yauhenbichel
 *
 */
public class RecordDuplicateException extends RuntimeException {

        private static final Logger logger = Logger.getLogger(
            RecordNotFoundException.class);

    public RecordDuplicateException(String errorMessage) {
        super(errorMessage);
        logger.error(errorMessage);
    }
    public RecordDuplicateException(String errorMessage, Throwable err) {
        super(errorMessage, err);
```

```
        logger.error(errorMessage, err);
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.holidaysystem.common.exception;

import org.jboss.logging.Logger;

/**
 * Raises when the system does not contain required record
 * @author yauhenbichel
 *
 */
public class RecordNotFoundException extends RuntimeException {

    private static final Logger logger = Logger.getLogger(
        RecordNotFoundException.class);

    public RecordNotFoundException(String errorMessage) {
        super(errorMessage);
        logger.error(errorMessage);
    }
    public RecordNotFoundException(String errorMessage, Throwable err) {
        super(errorMessage, err);
        logger.error(errorMessage, err);
    }
}
```

package com.holidaysystem.common.

```
/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
```

```
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.common;

import java.io.IOException;

import javax.ws.rs.container.ContainerRequestContext;
import javax.ws.rs.container.ContainerResponseContext;
import javax.ws.rs.container.ContainerResponseFilter;
import javax.ws.rs.ext.Provider;

/**
 *
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
//@Provider
public class CorsFilter implements ContainerResponseFilter {

    @Override
    public void filter(ContainerRequestContext requestContext,
                       ContainerResponseContext responseContext) throws
                          IOException {
        responseContext.getHeaders()
                .add("Access-Control-Allow-Origin", "*");
        responseContext.getHeaders()
          .add("Access-Control-Allow-Credentials", "true");
        responseContext.getHeaders()
          .add("Access-Control-Allow-Headers", "*");
        responseContext.getHeaders()
          .add("Access-Control-Allow-Methods", "*");
    }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
```

```
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.common;

import java.time.format.DateTimeFormatter;
import java.time.format.DateTimeFormatterBuilder;
import java.time.temporal.ChronoField;

/**
 * Utils for dates and time
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public final class DateUtils {
        public static final String BASE_PATTERN = "yyyy-MM-dd HH:mm:ss";
        public static final DateTimeFormatter FORMATTER =
                    new DateTimeFormatterBuilder().appendPattern(BASE_PATTERN)
                        .appendFraction(ChronoField.NANO_OF_SECOND, 0, 9, true
                            ).toFormatter();
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.common;

import com.holidaysystem.vo.RegistrationRequest;

/**
 * Builder for RegistrationRequest
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public final class RegistrationRequestBuilder {

        private static RegistrationRequestBuilder builder;
    private RegistrationRequest request;
```

```
    private RegistrationRequestBuilder() {
        request = new RegistrationRequest();
    }

    public static RegistrationRequestBuilder builder() {
        builder = new RegistrationRequestBuilder();
        return builder;
    }

    public RegistrationRequestBuilder setEmail(String email) {
        if(request == null) {
                request = new RegistrationRequest();
        }
        this.request.setEmail(email);
        return this;
    }

    public RegistrationRequestBuilder setPassword(String password) {
        if(request == null) {
                request = new RegistrationRequest();
        }
        this.request.setPassword(password);
        return this;
    }

    public RegistrationRequestBuilder setAuthRole(String authRole) {
        if(request == null) {
                request = new RegistrationRequest();
        }
        this.request.setAuthRole(authRole);
        return this;
    }

    public RegistrationRequest build() {
        return this.request;
    }
}
]


/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
```

```
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.common.exception;

import javax.ws.rs.core.Response;
import javax.ws.rs.ext.ExceptionMapper;
import javax.ws.rs.ext.Provider;
import org.jboss.logging.Logger;

/**
 * Handle exception and return user friendly message about error in server
    side
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Provider
public class UncaughtExceptionMapper implements ExceptionMapper<Exception> {

    private static final Logger logger = Logger.getLogger(
       UncaughtExceptionMapper.class);

    @Override
    public Response toResponse(Exception ex) {
        logger.error(ex.getMessage(), ex);

        return Response
                        .status(Response.Status.INTERNAL_SERVER_ERROR)
                        .entity(ex.getMessage())
                        .type("text/plain")
                        .build();
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.holidaysystem.common.exception;
```

```
import org.jboss.logging.Logger;

/**
 * Raises when the system contains the same records
 * @author yauhenbichel
 *
 */
public class RecordDuplicateException extends RuntimeException {

        private static final Logger logger = Logger.getLogger(
            RecordNotFoundException.class);

    public RecordDuplicateException(String errorMessage) {
        super(errorMessage);
        logger.error(errorMessage);
    }
    public RecordDuplicateException(String errorMessage, Throwable err) {
        super(errorMessage, err);
        logger.error(errorMessage, err);
    }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.holidaysystem.common.exception;

import org.jboss.logging.Logger;

/**
 * Raises when the system does not contain required record
 * @author yauhenbichel
 *
 */
public class RecordNotFoundException extends RuntimeException {

    private static final Logger logger = Logger.getLogger(
        RecordNotFoundException.class);
```

```
    public RecordNotFoundException(String errorMessage) {
        super(errorMessage);
        logger.error(errorMessage);
    }
    public RecordNotFoundException(String errorMessage, Throwable err) {
        super(errorMessage, err);
        logger.error(errorMessage, err);
    }
}
```

] package com.holidaysystem.common.exception;

```
/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.common.exception;

import javax.ws.rs.core.Response;
import javax.ws.rs.ext.ExceptionMapper;
import javax.ws.rs.ext.Provider;
import org.jboss.logging.Logger;

/**
 * Handle exception and return user friendly message about error in server
    side
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Provider
public class UncaughtExceptionMapper implements ExceptionMapper<Exception> {

    private static final Logger logger = Logger.getLogger(
        UncaughtExceptionMapper.class);

    @Override
    public Response toResponse(Exception ex) {
        logger.error(ex.getMessage(), ex);

        return Response
                        .status(Response.Status.INTERNAL_SERVER_ERROR)
                        .entity(ex.getMessage())
```

```
                        .type("text/plain")
                        .build();
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.holidaysystem.common.exception;

import org.jboss.logging.Logger;

/**
 * Raises when the system contains the same records
 * @author yauhenbichel
 *
 */
public class RecordDuplicateException extends RuntimeException {

        private static final Logger logger = Logger.getLogger(
            RecordNotFoundException.class);

    public RecordDuplicateException(String errorMessage) {
        super(errorMessage);
        logger.error(errorMessage);
    }
    public RecordDuplicateException(String errorMessage, Throwable err) {
        super(errorMessage, err);
        logger.error(errorMessage, err);
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
```

```
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.holidaysystem.common.exception;

import org.jboss.logging.Logger;

/**
 * Raises when the system does not contain required record
 * @author yauhenbichel
 *
 */
public class RecordNotFoundException extends RuntimeException {

    private static final Logger logger = Logger.getLogger(
        RecordNotFoundException.class);

    public RecordNotFoundException(String errorMessage) {
        super(errorMessage);
        logger.error(errorMessage);
    }
    public RecordNotFoundException(String errorMessage, Throwable err) {
        super(errorMessage, err);
        logger.error(errorMessage, err);
    }
}
```

package com.holidaysystem.common.

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.common;
```

```
import java.io.IOException;

import javax.ws.rs.container.ContainerRequestContext;
import javax.ws.rs.container.ContainerResponseContext;
import javax.ws.rs.container.ContainerResponseFilter;
import javax.ws.rs.ext.Provider;

/**
 *
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
//@Provider
public class CorsFilter implements ContainerResponseFilter {

    @Override
    public void filter(ContainerRequestContext requestContext,
                       ContainerResponseContext responseContext) throws
                          IOException {
        responseContext.getHeaders()
                .add("Access-Control-Allow-Origin", "*");
        responseContext.getHeaders()
          .add("Access-Control-Allow-Credentials", "true");
        responseContext.getHeaders()
          .add("Access-Control-Allow-Headers", "*");
        responseContext.getHeaders()
          .add("Access-Control-Allow-Methods", "*");
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.common;

import java.time.format.DateTimeFormatter;
import java.time.format.DateTimeFormatterBuilder;
import java.time.temporal.ChronoField;

/**
```

```
 * Utils for dates and time
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public final class DateUtils {
        public static final String BASE_PATTERN = "yyyy-MM-dd HH:mm:ss";
        public static final DateTimeFormatter FORMATTER =
                    new DateTimeFormatterBuilder().appendPattern(BASE_PATTERN)
                        .appendFraction(ChronoField.NANO_OF_SECOND, 0, 9, true
                            ).toFormatter();
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.common;

import com.holidaysystem.vo.RegistrationRequest;

/**
 * Builder for RegistrationRequest
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public final class RegistrationRequestBuilder {

        private static RegistrationRequestBuilder builder;
    private RegistrationRequest request;

    private RegistrationRequestBuilder() {
        request = new RegistrationRequest();
    }

    public static RegistrationRequestBuilder builder() {
        builder = new RegistrationRequestBuilder();
        return builder;
    }

    public RegistrationRequestBuilder setEmail(String email) {
        if(request == null) {
```

```
                request = new RegistrationRequest();
        }
        this.request.setEmail(email);
        return this;
    }

    public RegistrationRequestBuilder setPassword(String password) {
        if(request == null) {
                request = new RegistrationRequest();
        }
        this.request.setPassword(password);
        return this;
    }

    public RegistrationRequestBuilder setAuthRole(String authRole) {
        if(request == null) {
                request = new RegistrationRequest();
        }
        this.request.setAuthRole(authRole);
        return this;
    }

    public RegistrationRequest build() {
        return this.request;
    }
}



/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.comparator;

import java.util.Comparator;

import javax.enterprise.context.ApplicationScoped;

import com.google.common.collect.ComparisonChain;
import com.holidaysystem.model.HolidayRequestModel;
```

```
/**
 * Comparator for prioritization
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
public class HolidayRequestModelComparator implements Comparator<
    HolidayRequestModel> {

        @Override
        public int compare(HolidayRequestModel m1, HolidayRequestModel m2) {
                return ComparisonChain.start()
                .compare(m1.getTakenDays(), m2.getTakenDays())
                .compare(m1.getRequestedDays(), m2.getRequestedDays())
                .result();

        }

}
```

] package com.holidaysystem.comparator;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
     eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *           http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.comparator;

import java.util.Comparator;

import javax.enterprise.context.ApplicationScoped;

import com.google.common.collect.ComparisonChain;
import com.holidaysystem.model.HolidayRequestModel;

/**
 * Comparator for prioritization
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
public class HolidayRequestModelComparator implements Comparator<
```

```
    HolidayRequestModel> {

        @Override
        public int compare(HolidayRequestModel m1, HolidayRequestModel m2) {
                return ComparisonChain.start()
                .compare(m1.getTakenDays(), m2.getTakenDays())
                .compare(m1.getRequestedDays(), m2.getRequestedDays())
                .result();

        }

}


/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.config;

import javax.enterprise.context.ApplicationScoped;

@ApplicationScoped
public class AppConfig {

}
```

] package com.holidaysystem.config;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
```

```
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.config;

import javax.enterprise.context.ApplicationScoped;

@ApplicationScoped
public class AppConfig {

}



/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.constraints;

import java.util.List;

import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;

import com.holidaysystem.enumeration.EmployeeRoleEnum;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.model.EmployeeModel;

/**
 *
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class ConstraintsValidator implements IConstraintsValidator {

        @Override
        public boolean hasHeadOrDeputyHead(List<EmployeeModel> epmployees) {
                return epmployees.stream().filter(employee ->
```

```
                        employee.getRole().equals(EmployeeRoleEnum.HEAD) ||
                        employee.getRole().equals(EmployeeRoleEnum.DEPUTY_HEAD
                            ))
                                .count() > 0;
        }

        @Override
        public boolean hasManagerAndSeniorMemeber(List<EmployeeModel>
            epmployees) {
                return epmployees.stream().filter(employee ->
                        employee.getRole().equals(EmployeeRoleEnum.MANAGER) &&
                        employee.getRole().equals(EmployeeRoleEnum.
                            SENIOR_MEMBER))
                                .count() > 0;
        }

        @Override
        public boolean hasSpecificPercentMemebersOnDuty(List<EmployeeModel>
            epmployees, Integer percentOnDuty) {
                long numsOnDuty = epmployees.stream().filter(employee ->
                        employee.getHolidayStatus().equals(HolidayStatusEnum.
                            ON_DUTY))
                                .count();

                int numsOfAll = epmployees.size();
                final int oneHundrenPercent = 100;

                return (numsOnDuty * oneHundrenPercent / numsOfAll) >=
                    percentOnDuty ? true: false;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.constraints;

import java.util.List;

import com.holidaysystem.model.EmployeeModel;
```

```
/**
 *
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IConstraintsValidator {
        /**
         *
         * @param epmployees
         * @return
         */
        boolean hasHeadOrDeputyHead(List<EmployeeModel> epmployees);
        /**
         *
         * @param epmployees
         * @return
         */
        boolean hasManagerAndSeniorMemeber(List<EmployeeModel> epmployees);
        /**
         *
         * @param epmployees
         * @param percentOnDuty
         * @return
         */
        boolean hasSpecificPercentMemebersOnDuty(List<EmployeeModel>
            epmployees, Integer percentOnDuty);
}
```

] package com.holidaysystem.constraints;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.constraints;

import java.util.List;

import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
```

```
import com.holidaysystem.enumeration.EmployeeRoleEnum;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.model.EmployeeModel;

/**
 *
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class ConstraintsValidator implements IConstraintsValidator {

        @Override
        public boolean hasHeadOrDeputyHead(List<EmployeeModel> epmployees) {
                return epmployees.stream().filter(employee ->
                        employee.getRole().equals(EmployeeRoleEnum.HEAD) ||
                        employee.getRole().equals(EmployeeRoleEnum.DEPUTY_HEAD
                            ))
                                .count() > 0;
        }

        @Override
        public boolean hasManagerAndSeniorMemeber(List<EmployeeModel>
            epmployees) {
                return epmployees.stream().filter(employee ->
                        employee.getRole().equals(EmployeeRoleEnum.MANAGER) &&
                        employee.getRole().equals(EmployeeRoleEnum.
                            SENIOR_MEMBER))
                                .count() > 0;
        }

        @Override
        public boolean hasSpecificPercentMemebersOnDuty(List<EmployeeModel>
            epmployees, Integer percentOnDuty) {
                long numsOnDuty = epmployees.stream().filter(employee ->
                        employee.getHolidayStatus().equals(HolidayStatusEnum.
                            ON_DUTY))
                                .count();

                int numsOfAll = epmployees.size();
                final int oneHundrenPercent = 100;

                return (numsOnDuty * oneHundrenPercent / numsOfAll) >=
                    percentOnDuty ? true: false;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
```

```
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.constraints;

import java.util.List;

import com.holidaysystem.model.EmployeeModel;

/**
 *
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IConstraintsValidator {
        /**
         *
         * @param epmployees
         * @return
         */
        boolean hasHeadOrDeputyHead(List<EmployeeModel> epmployees);
        /**
         *
         * @param epmployees
         * @return
         */
        boolean hasManagerAndSeniorMemeber(List<EmployeeModel> epmployees);
        /**
         *
         * @param epmployees
         * @param percentOnDuty
         * @return
         */
        boolean hasSpecificPercentMemebersOnDuty(List<EmployeeModel>
            epmployees, Integer percentOnDuty);
}


/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
```

```
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import java.util.UUID;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.Email;
import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;

/**
 * Account entity
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Entity
@Table(name = "account", schema = "public")
public class AccountEntity extends AuditEntity {

        private static final long serialVersionUID = 1L;

        @Id
        @Column(name = "id")
        @GeneratedValue(strategy = GenerationType.IDENTITY)
    private UUID id;

        @Column(name = "auth_roleid")
        private UUID authRoleId;

    @Email
    @Column(name = "email")
    private String email;

    @NotBlank
    @Column(name = "password")
    private String password;

    @NotNull
    @Column(name = "active")
    private Boolean active;
```

```
    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setEmail(String email) {this.email = email; }
    public String getEmail() {return this.email;}
    public void setPassword(String password) {this.password = password; }
    public String getPassword() {return this.password;}
    public void setAuthRoleId(UUID authRoleId) {this.authRoleId = authRoleId;
        }
    public UUID getAuthRoleId() {return this.authRoleId;}
    public void setActive(Boolean active) {this.active = active; }
    public Boolean getActive() {return this.active;}
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import javax.persistence.Column;
import javax.persistence.MappedSuperclass;
import javax.validation.constraints.PastOrPresent;

import java.time.LocalDateTime;

/**
 * Entity with created and modified timestamps for tracing changes
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@MappedSuperclass
public class AuditEntity {

    private static final long serialVersionUID = 1L;

    @PastOrPresent
    @Column(name = "created")
    private LocalDateTime created;

    @PastOrPresent
    @Column(name = "modified")
```

```
    private LocalDateTime modified;

    public void setCreated(LocalDateTime created) {
        this.created = created;
    }
    public LocalDateTime getCreated() {
        return this.created;
    }
    public void setModified(LocalDateTime modified) {
        this.modified = modified;
    }
    public LocalDateTime getModified() {
        return this.modified;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import java.util.UUID;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.Email;
import javax.validation.constraints.NotBlank;

/**
 * AuthorizationRole entity
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Entity
@Table(name = "authorization_role", schema = "public")
public class AuthorizationRoleEntity {
```

```
        private static final long serialVersionUID = 1L;

        @Id
        @Column(name = "id")
        @GeneratedValue(strategy = GenerationType.IDENTITY)
    private UUID id;

        @NotBlank
    @Column(name = "name")
    private String name;

    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setName(String name) {this.name = name; }
    public String getName() {return this.name;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import java.util.UUID;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.NotBlank;
import javax.validation.constraints.PositiveOrZero;

/**
 * Employee entity
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Entity
```

```
@Table(name = "employee", schema = "public")
public class EmployeeEntity extends AuditEntity {

        @Id
        @Column(name = "id")
        @GeneratedValue(strategy = GenerationType.IDENTITY)
    private UUID id;

        @Column(name = "acountid")
    private UUID accountId;

        @NotBlank
        @Column(name = "role")
        private String role;

        @NotBlank
        @Column(name = "department")
        private String department;

        @NotBlank
        @Column(name = "firstname")
    private String firstName;

        @NotBlank
        @Column(name = "lastname")
        private String lastName;

        @PositiveOrZero
        @Column(name = "years")
    private Integer years;

    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setDepartment(String department) {this.department = department
        ; }
    public String getDepartment() {return this.department;}
    public void setRole(String role) {this.role = role; }
    public String getRole() {return this.role;}
    public void setAccountId(UUID accountId) {this.accountId = accountId; }
    public UUID getAccountId() {return this.accountId;}
    public void setFirstName(String firstName) {this.firstName = firstName; }
    public String getFirstName() {return this.firstName;}
    public void setLastName(String lastName) {this.lastName = lastName; }
    public String getLastName() {return this.lastName;}
    public void setYears(Integer years) {this.years = years; }
    public Integer getYears() {return this.years;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
```

```
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import java.util.UUID;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.Positive;
import javax.validation.constraints.PositiveOrZero;

/**
 * HolidayDetails entity
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Entity
@Table(name = "holiday_details", schema = "public")
public class HolidayDetailsEntity extends AuditEntity {

        private static final long serialVersionUID = 1L;

        @Id
        @Column(name = "id")
        @GeneratedValue(strategy = GenerationType.IDENTITY)
    private UUID id;

        @Column(name = "employeeid")
    private UUID employeeId;

        @Positive
        @Column(name = "totaldays")
        private Integer totalDays;

        @PositiveOrZero
        @Column(name = "takendays")
        private Integer takenDays;

        @Column(name = "status")
        private String status;
```

```
    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setEmployeeId(UUID employeeId) {this.employeeId = employeeId;
        }
    public UUID getEmployeeId() {return this.employeeId;}
    public void setTotalDays(Integer totalDays) {this.totalDays = totalDays; }
    public Integer getTotalDays() {return this.totalDays;}
    public void setTakenDays(Integer takenDays) {this.takenDays = takenDays; }
    public Integer getTakenDays() {return this.takenDays;}
    public void setStatus(String status) {this.status = status; }
    public String getStatus() {return this.status;}
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import java.time.LocalDateTime;
import java.util.UUID;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.FutureOrPresent;


/**
 * HolidayRequest entity
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Entity
@Table(name = "holiday-request", schema = "public")
public class HolidayRequestEntity extends AuditEntity {
        private static final long serialVersionUID = 1L;
```

```
        @Id
        @Column(name = "id")
        @GeneratedValue(strategy = GenerationType.IDENTITY)
    private UUID id;

        @Column(name = "employeeid")
    private UUID employeeId;

        @FutureOrPresent
        @Column(name = "startdate")
        private LocalDateTime startDate;

        @FutureOrPresent
        @Column(name = "enddate")
        private LocalDateTime endDate;

        @Column(name = "status")
        private String status;

    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setEmployeeId(UUID employeeId) {this.employeeId = employeeId;
       }
    public UUID getEmployeeId() {return this.employeeId;}
    public void setStatus(String status) {this.status = status; }
    public String getStatus() {return this.status;}
    public void setStartDate(LocalDateTime startDate) {this.startDate =
       startDate; }
    public LocalDateTime getStartDate() {return this.startDate;}
    public void setEndDate(LocalDateTime endDate) {this.endDate = endDate; }
    public LocalDateTime getEndDate() {return this.endDate;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import java.time.LocalDateTime;
import java.util.UUID;
```

```
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.FutureOrPresent;

/**
 * RequestAlert entity
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Entity
@Table(name = "request_alert_queue", schema = "public")
public class RequestAlertEntity extends AuditEntity {
        private static final long serialVersionUID = 1L;

        @Id
        @Column(name = "id")
        @GeneratedValue(strategy = GenerationType.IDENTITY)
    private UUID id;
        @Column(name = "requestid")
    private UUID requestId;

        @FutureOrPresent
        @Column(name = "date")
        private LocalDateTime date;

    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setRequestId(UUID requestId) {this.requestId = requestId; }
    public UUID getRequestId() {return this.requestId;}
    public void setDate(LocalDateTime date) {this.date = date; }
    public LocalDateTime getDate() {return this.date;}
}
```

] package com.holidaysystem.entity;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
```

```
 * limitations under the License.
 */
package com.holidaysystem.entity;

import java.util.UUID;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.Email;
import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;

/**
 * Account entity
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Entity
@Table(name = "account", schema = "public")
public class AccountEntity extends AuditEntity {

        private static final long serialVersionUID = 1L;

        @Id
        @Column(name = "id")
        @GeneratedValue(strategy = GenerationType.IDENTITY)
    private UUID id;

        @Column(name = "auth_roleid")
        private UUID authRoleId;

    @Email
    @Column(name = "email")
    private String email;

    @NotBlank
    @Column(name = "password")
    private String password;

    @NotNull
    @Column(name = "active")
    private Boolean active;

    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setEmail(String email) {this.email = email; }
    public String getEmail() {return this.email;}
    public void setPassword(String password) {this.password = password; }
    public String getPassword() {return this.password;}
    public void setAuthRoleId(UUID authRoleId) {this.authRoleId = authRoleId;
```

```
        }
    public UUID getAuthRoleId() {return this.authRoleId;}
    public void setActive(Boolean active) {this.active = active; }
    public Boolean getActive() {return this.active;}
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import javax.persistence.Column;
import javax.persistence.MappedSuperclass;
import javax.validation.constraints.PastOrPresent;

import java.time.LocalDateTime;

/**
 * Entity with created and modified timestamps for tracing changes
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@MappedSuperclass
public class AuditEntity {

    private static final long serialVersionUID = 1L;

    @PastOrPresent
    @Column(name = "created")
    private LocalDateTime created;

    @PastOrPresent
    @Column(name = "modified")
    private LocalDateTime modified;

    public void setCreated(LocalDateTime created) {
        this.created = created;
    }
    public LocalDateTime getCreated() {
        return this.created;
```

```
    }
    public void setModified(LocalDateTime modified) {
        this.modified = modified;
    }
    public LocalDateTime getModified() {
        return this.modified;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import java.util.UUID;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.Email;
import javax.validation.constraints.NotBlank;

/**
 * AuthorizationRole entity
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Entity
@Table(name = "authorization_role", schema = "public")
public class AuthorizationRoleEntity {

        private static final long serialVersionUID = 1L;

        @Id
        @Column(name = "id")
        @GeneratedValue(strategy = GenerationType.IDENTITY)
    private UUID id;
```

```
        @NotBlank
    @Column(name = "name")
    private String name;

    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setName(String name) {this.name = name; }
    public String getName() {return this.name;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import java.util.UUID;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.NotBlank;
import javax.validation.constraints.PositiveOrZero;

/**
 * Employee entity
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Entity
@Table(name = "employee", schema = "public")
public class EmployeeEntity extends AuditEntity {

        @Id
        @Column(name = "id")
        @GeneratedValue(strategy = GenerationType.IDENTITY)
    private UUID id;
```

```
        @Column(name = "acountid")
    private UUID accountId;

        @NotBlank
        @Column(name = "role")
        private String role;

        @NotBlank
        @Column(name = "department")
        private String department;

        @NotBlank
        @Column(name = "firstname")
    private String firstName;

        @NotBlank
        @Column(name = "lastname")
        private String lastName;

        @PositiveOrZero
        @Column(name = "years")
    private Integer years;

    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setDepartment(String department) {this.department = department
        ; }
    public String getDepartment() {return this.department;}
    public void setRole(String role) {this.role = role; }
    public String getRole() {return this.role;}
    public void setAccountId(UUID accountId) {this.accountId = accountId; }
    public UUID getAccountId() {return this.accountId;}
    public void setFirstName(String firstName) {this.firstName = firstName; }
    public String getFirstName() {return this.firstName;}
    public void setLastName(String lastName) {this.lastName = lastName; }
    public String getLastName() {return this.lastName;}
    public void setYears(Integer years) {this.years = years; }
    public Integer getYears() {return this.years;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
```

```
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import java.util.UUID;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.Positive;
import javax.validation.constraints.PositiveOrZero;

/**
 * HolidayDetails entity
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Entity
@Table(name = "holiday_details", schema = "public")
public class HolidayDetailsEntity extends AuditEntity {

        private static final long serialVersionUID = 1L;

        @Id
        @Column(name = "id")
        @GeneratedValue(strategy = GenerationType.IDENTITY)
    private UUID id;

        @Column(name = "employeeid")
    private UUID employeeId;

        @Positive
        @Column(name = "totaldays")
        private Integer totalDays;

        @PositiveOrZero
        @Column(name = "takendays")
        private Integer takenDays;

        @Column(name = "status")
        private String status;

    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setEmployeeId(UUID employeeId) {this.employeeId = employeeId;
       }
    public UUID getEmployeeId() {return this.employeeId;}
    public void setTotalDays(Integer totalDays) {this.totalDays = totalDays; }
    public Integer getTotalDays() {return this.totalDays;}
```

```
    public void setTakenDays(Integer takenDays) {this.takenDays = takenDays; }
    public Integer getTakenDays() {return this.takenDays;}
    public void setStatus(String status) {this.status = status; }
    public String getStatus() {return this.status;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import java.time.LocalDateTime;
import java.util.UUID;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.validation.constraints.FutureOrPresent;


/**
 * HolidayRequest entity
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Entity
@Table(name = "holiday-request", schema = "public")
public class HolidayRequestEntity extends AuditEntity {
        private static final long serialVersionUID = 1L;

        @Id
        @Column(name = "id")
        @GeneratedValue(strategy = GenerationType.IDENTITY)
    private UUID id;

        @Column(name = "employeeid")
    private UUID employeeId;
```

```
        @FutureOrPresent
        @Column(name = "startdate")
        private LocalDateTime startDate;

        @FutureOrPresent
        @Column(name = "enddate")
        private LocalDateTime endDate;

        @Column(name = "status")
        private String status;

    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setEmployeeId(UUID employeeId) {this.employeeId = employeeId;
        }
    public UUID getEmployeeId() {return this.employeeId;}
    public void setStatus(String status) {this.status = status; }
    public String getStatus() {return this.status;}
    public void setStartDate(LocalDateTime startDate) {this.startDate =
        startDate; }
    public LocalDateTime getStartDate() {return this.startDate;}
    public void setEndDate(LocalDateTime endDate) {this.endDate = endDate; }
    public LocalDateTime getEndDate() {return this.endDate;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.entity;

import java.time.LocalDateTime;
import java.util.UUID;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
```

```
import javax.validation.constraints.FutureOrPresent;

/**
 * RequestAlert entity
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Entity
@Table(name = "request_alert_queue", schema = "public")
public class RequestAlertEntity extends AuditEntity {
        private static final long serialVersionUID = 1L;

        @Id
        @Column(name = "id")
        @GeneratedValue(strategy = GenerationType.IDENTITY)
    private UUID id;
        @Column(name = "requestid")
    private UUID requestId;

        @FutureOrPresent
        @Column(name = "date")
        private LocalDateTime date;

    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setRequestId(UUID requestId) {this.requestId = requestId; }
    public UUID getRequestId() {return this.requestId;}
    public void setDate(LocalDateTime date) {this.date = date; }
    public LocalDateTime getDate() {return this.date;}
}


/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.enumeration;

/**
 * Authorization Role Enum
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
```

```
 *
 */
public enum AuthorizationRoleEnum {
        USER,
        ADMIN
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.enumeration;

/**
 * Department Enum
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public enum DepartmentEnum {
        ENGINEERING,
        PLUMBING,
        ROOFING,
        OFFICE,
        BRICKLAYING,
        CARPENTLY
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
```

```
 * limitations under the License.
 */
package com.holidaysystem.enumeration;

/**
 * Employee Role Enum
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public enum EmployeeRoleEnum {
        HEAD,
        DEPUTY_HEAD,
        MANAGER,
        APPRENTICE,
        JUNIOR_MEMBER,
        SENIOR_MEMBER
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.enumeration;

/**
 * Holiday Request Status Enum
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public enum HolidayRequestStatusEnum {
        PENDING,
        APPROVED,
        DECLINED,
        BROKEN
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
```

```
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.enumeration;

/**
 * Holiday Status Enum
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public enum HolidayStatusEnum {
        ON_DUTY,
        ON_HOLIDAY
}
```

] package com.holidaysystem.enumeration;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.enumeration;

/**
 * Authorization Role Enum
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public enum AuthorizationRoleEnum {
        USER,
        ADMIN
}

/*
```

```
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.enumeration;

/**
 * Department Enum
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public enum DepartmentEnum {
        ENGINEERING,
        PLUMBING,
        ROOFING,
        OFFICE,
        BRICKLAYING,
        CARPENTLY
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.enumeration;

/**
 * Employee Role Enum
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
```

```
 *
 */
public enum EmployeeRoleEnum {
        HEAD,
        DEPUTY_HEAD,
        MANAGER,
        APPRENTICE,
        JUNIOR_MEMBER,
        SENIOR_MEMBER
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.enumeration;

/**
 * Holiday Request Status Enum
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public enum HolidayRequestStatusEnum {
        PENDING,
        APPROVED,
        DECLINED,
        BROKEN
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
```

```
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.enumeration;

/**
 * Holiday Status Enum
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public enum HolidayStatusEnum {
        ON_DUTY,
        ON_HOLIDAY
}



/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.mail;

/**
 *
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IMailService {
        /**
         *
         */
        void send();
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
```

```
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.mail;

import java.util.Properties;

import javax.enterprise.context.ApplicationScoped;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

import org.jboss.logging.Logger;

import jakarta.inject.Named;

/**
 * Mail service for sending message to admin
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Named("DeafultMailService")
public class MailService implements IMailService {

        private static final Logger logger = Logger.getLogger(MailService.
           class);

        public void send() {
              //provide recipient's email ID
              String to = "yb3129h@gre.ac.uk";

              String from = "help.holiday.request@gmail.com";
              final String username = "Holiday Request System";
              final String password = "1Vaction2!Request%";


              Properties prop = new Properties();
                prop.put("mail.smtp.host", "smtp.gmail.com");
                prop.put("mail.smtp.port", "587");
                prop.put("mail.smtp.auth", "true");
                prop.put("mail.smtp.starttls.enable", "true");
```

```
                prop.put("mail.smtp.starttls.required", "true");

                Session session = Session.getInstance(prop,
                        new javax.mail.Authenticator() {
                            protected PasswordAuthentication
                                getPasswordAuthentication() {
                                return new PasswordAuthentication(from,
                                    password);
                            }
                        });

                try {

                    Message message = new MimeMessage(session);
                    message.setFrom(new InternetAddress(from));
                    message.setRecipients(Message.RecipientType.TO,
                                          InternetAddress.parse(to));

                    message.setSubject("COMP1610 Group 5 Notification for
                        Admin!");
                            message.setText("There is a new request!");

                            Transport.send(message);

                    logger.debug("Email Message Sent Successfully");

                } catch (MessagingException e) {
                    e.printStackTrace();
                    logger.error(e.getMessage(), e);
                }
            }
}
```

] package com.holidaysystem.mail;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *           http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.mail;

/**
```

```
 *
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IMailService {
        /**
         *
         */
        void send();
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.mail;

import java.util.Properties;

import javax.enterprise.context.ApplicationScoped;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

import org.jboss.logging.Logger;

import jakarta.inject.Named;

/**
 * Mail service for sending message to admin
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Named("DeafultMailService")
public class MailService implements IMailService {
```

```
        private static final Logger logger = Logger.getLogger(MailService.
            class);

        public void send() {
              //provide recipient's email ID
              String to = "yb3129h@gre.ac.uk";

              String from = "help.holiday.request@gmail.com";
              final String username = "Holiday Request System";
              final String password = "1Vaction2!Request%";


              Properties prop = new Properties();
                prop.put("mail.smtp.host", "smtp.gmail.com");
                prop.put("mail.smtp.port", "587");
                prop.put("mail.smtp.auth", "true");
                prop.put("mail.smtp.starttls.enable", "true");
                prop.put("mail.smtp.starttls.required", "true");

                Session session = Session.getInstance(prop,
                        new javax.mail.Authenticator() {
                            protected PasswordAuthentication
                                getPasswordAuthentication() {
                                return new PasswordAuthentication(from,
                                    password);
                            }
                        });

                try {

                    Message message = new MimeMessage(session);
                    message.setFrom(new InternetAddress(from));
                    message.setRecipients(Message.RecipientType.TO,
                                          InternetAddress.parse(to));

                    message.setSubject("COMP1610 Group 5 Notification for
                        Admin!");
                          message.setText("There is a new request!");

                          Transport.send(message);

                    logger.debug("Email Message Sent Successfully");

                } catch (MessagingException e) {
                    e.printStackTrace();
                    logger.error(e.getMessage(), e);
                }
            }
}


/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
```

```
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.mapper;

import java.time.LocalDateTime;
import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;

import com.holidaysystem.entity.AccountEntity;
import com.holidaysystem.entity.AuthorizationRoleEntity;
import com.holidaysystem.enumeration.AuthorizationRoleEnum;
import com.holidaysystem.model.AccountDetailsModel;
import com.holidaysystem.vo.AccountResponse;
import com.holidaysystem.vo.RegistrationRequest;

/**
 * Account mapper
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
public class AccountMapper {

        public AccountEntity toEntity(UUID id, RegistrationRequest
            registrationRequest, String hashedPassWithSalt) {
                AccountEntity account = new AccountEntity();
        account.setId(id);
        account.setEmail(registrationRequest.getEmail());
        account.setPassword(hashedPassWithSalt);
        account.setActive(true);
        account.setCreated(LocalDateTime.now());
                account.setModified(LocalDateTime.now());

        return account;
        }

        public AccountResponse toResponse(AccountDetailsModel model) {
                AccountResponse resp = new AccountResponse();
        resp.setId(model.getId());
        resp.setEmail(model.getEmail());
        resp.setActive(model.getActive());
```

```
        resp.setAuthRole(model.getAuthRole().name());

                return resp;
        }

        public AccountDetailsModel toModel(AccountEntity account,
            AuthorizationRoleEntity authRole) {
                AccountDetailsModel accountDetails = new AccountDetailsModel()
                    ;
        accountDetails.setId(account.getId());
        accountDetails.setEmail(account.getEmail());
        accountDetails.setActive(account.getActive());
        accountDetails.setAuthRole(AuthorizationRoleEnum.valueOf(authRole.
            getName()));

        return accountDetails;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.mapper;

import java.time.LocalDateTime;
import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;

import com.holidaysystem.entity.AccountEntity;
import com.holidaysystem.entity.EmployeeEntity;
import com.holidaysystem.entity.HolidayDetailsEntity;
import com.holidaysystem.enumeration.DepartmentEnum;
import com.holidaysystem.model.EmployeeModel;
import com.holidaysystem.enumeration.EmployeeRoleEnum;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.vo.NewEmployeeRequest;
import com.holidaysystem.vo.EmployeeResponse;

/**
```

```
 * Employee mapper
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
public class EmployeeMapper {
        public EmployeeEntity toEntity(NewEmployeeRequest request) {
                EmployeeEntity entity = new EmployeeEntity();
        entity.setId(UUID.randomUUID());
        entity.setAccountId(request.getAccountId());
        entity.setRole(request.getRole());
        entity.setDepartment(request.getDepartment());
        entity.setFirstName(request.getFirstName());
        entity.setLastName(request.getLastName());
        entity.setYears(request.getYears());
        entity.setCreated(LocalDateTime.now());
        entity.setModified(LocalDateTime.now());

        return entity;
        }

        public EmployeeResponse toResponse(EmployeeEntity entity) {
                EmployeeResponse employee = new EmployeeResponse();
                employee.setId(entity.getId());
                employee.setAccountId(entity.getAccountId());
                employee.setFirstName(entity.getFirstName());
                employee.setLastName(entity.getLastName());
                employee.setRole(entity.getRole());
                employee.setDepartment(entity.getDepartment());
                employee.setYears(entity.getYears());

                return employee;
        }

        public EmployeeModel toEmployeeModel(EmployeeEntity entity) {
                EmployeeModel employee = new EmployeeModel();
                employee.setId(entity.getId());
                employee.setAccountId(entity.getAccountId());
                employee.setFirstName(entity.getFirstName());
                employee.setLastName(entity.getLastName());
                employee.setRole(EmployeeRoleEnum.valueOf(entity.getRole()));
                employee.setDepartment(DepartmentEnum.valueOf(entity.
                    getDepartment()));
                employee.setYears(entity.getYears());

                return employee;
        }

        public EmployeeModel toEmployeeModel(EmployeeEntity employeeEntity,
            AccountEntity accountEntity, HolidayDetailsEntity
            holidayDetailsEntity) {
                EmployeeModel employee = new EmployeeModel();
                employee.setId(employeeEntity.getId());
                employee.setAccountId(employeeEntity.getAccountId());
```

```
                employee.setEmail(accountEntity.getEmail());
                employee.setFirstName(employeeEntity.getFirstName());
                employee.setLastName(employeeEntity.getLastName());
                employee.setRole(EmployeeRoleEnum.valueOf(employeeEntity.
                    getRole()));
                employee.setDepartment(DepartmentEnum.valueOf(employeeEntity.
                    getDepartment()));
                employee.setYears(employeeEntity.getYears());
                employee.setTotalDays(holidayDetailsEntity.getTotalDays());
                employee.setTakenDays(holidayDetailsEntity.getTakenDays());

                return employee;
        }

        public EmployeeResponse toResponse(EmployeeModel model) {
                EmployeeResponse employee = new EmployeeResponse();
                employee.setId(model.getId());
                employee.setAccountId(model.getAccountId());
                employee.setEmail(model.getEmail());
                employee.setFirstName(model.getFirstName());
                employee.setLastName(model.getLastName());
                employee.setRole(model.getRole().name());
                employee.setDepartment(model.getDepartment().name());
                employee.setYears(model.getYears());
                employee.setTotalDays(model.getTotalDays());
                employee.setTakenDays(model.getTakenDays());
                employee.setHolidayStatus(model.getHolidayStatus() == null?
                    HolidayStatusEnum.ON_DUTY.name()
                                : model.getHolidayStatus().name());

                return employee;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.mapper;

import java.time.LocalDateTime;
```

```
import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;

import com.fasterxml.jackson.core.JsonGenerationException;
import com.fasterxml.jackson.core.JsonProcessingException;
import com.fasterxml.jackson.databind.JsonMappingException;
import com.fasterxml.jackson.databind.ObjectMapper;
import com.fasterxml.jackson.databind.ObjectWriter;
import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.HolidayRequestEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.message.HolidayRequestMessage;
import com.holidaysystem.model.HolidayRequestModel;
import com.holidaysystem.vo.HolidayRequest;
import com.holidaysystem.vo.HolidayResponse;

/**
 * Holiday Request mapper
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
public class HolidayRequestMapper {
        public HolidayRequestEntity toEntity(UUID id, HolidayRequest request)
            {
                HolidayRequestEntity entity = new HolidayRequestEntity();
        entity.setId(id);
        entity.setEmployeeId(request.getEmployeeId());
        entity.setStatus(request.getStatus());
        entity.setStartDate(LocalDateTime.parse(request.getStartDate(),
            DateUtils.FORMATTER));
        entity.setEndDate(LocalDateTime.parse(request.getEndDate(), DateUtils.
            FORMATTER));
        entity.setCreated(LocalDateTime.now());
        entity.setModified(LocalDateTime.now());

        return entity;
        }

        public HolidayRequestEntity toEntity(HolidayRequestMessage
            holidayRequestMessage) {
                HolidayRequestEntity entity = new HolidayRequestEntity();
        entity.setId(holidayRequestMessage.getId());
        entity.setEmployeeId(holidayRequestMessage.getEmployeeId());
        entity.setStatus(holidayRequestMessage.getStatus());
        entity.setStartDate(LocalDateTime.parse(holidayRequestMessage.
            getStartDate(), DateUtils.FORMATTER));
        entity.setEndDate(LocalDateTime.parse(holidayRequestMessage.getEndDate
            (), DateUtils.FORMATTER));
        entity.setCreated(LocalDateTime.now());
        entity.setModified(LocalDateTime.now());

        return entity;
```

```
}

public HolidayRequestMessage toMessage(UUID id, HolidayRequest request
    ) {
        HolidayRequestMessage message = new HolidayRequestMessage();
        message.setId(id);
        message.setEmployeeId(request.getEmployeeId());
        message.setStatus(request.getStatus());
        message.setStartDate(request.getStartDate());
        message.setEndDate(request.getEndDate());

return message;
}

public HolidayRequestMessage toMessage(String jsonMessage) {

try {
        ObjectMapper objectMapper = new ObjectMapper();
        HolidayRequestMessage message = objectMapper.readValue(
            jsonMessage, HolidayRequestMessage.class);

    return message;
}
catch (JsonGenerationException | JsonMappingException  e) {
    e.printStackTrace();
}
catch(JsonProcessingException e) {
        e.printStackTrace();
}
catch(Exception e) {
        e.printStackTrace();
}

return null;
}

public String toJson(UUID id, HolidayRequest request) {
        HolidayRequestMessage message = toMessage(id, request);


try {
        ObjectWriter ow = new ObjectMapper().writer().
            withDefaultPrettyPrinter();
        String json = ow.writeValueAsString(message);

    return json;
}
catch (JsonGenerationException | JsonMappingException  e) {
    e.printStackTrace();
}
catch(JsonProcessingException e) {
        e.printStackTrace();
}
catch(Exception e) {
```

```
        e.printStackTrace();
}

return null;
}

public HolidayRequestEntity toEntity(HolidayRequest updated,
    HolidayRequestEntity db) {
        HolidayRequestEntity entity = new HolidayRequestEntity();
entity.setId(db.getId());
entity.setEmployeeId(updated.getEmployeeId());
entity.setStatus(updated.getStatus());
entity.setStartDate(LocalDateTime.parse(updated.getStartDate(),
    DateUtils.FORMATTER));
entity.setEndDate(LocalDateTime.parse(updated.getEndDate(), DateUtils.
   FORMATTER));
entity.setCreated(db.getCreated());
entity.setModified(LocalDateTime.now());

return entity;
}

public HolidayResponse toResponse(HolidayRequestModel model) {
        HolidayResponse holidayResponse = new HolidayResponse();
        holidayResponse.setId(model.getId());
        holidayResponse.setEmployeeId(model.getEmployeeId());
        holidayResponse.setStatus(model.getRequestStatus().name());
        holidayResponse.setStartDate(model.getStartDate().toString());
        holidayResponse.setEndDate(model.getEndDate().toString());
        holidayResponse.setTakenDays(model.getTakenDays());
        holidayResponse.setTotalDays(model.getTotalDays());
        holidayResponse.setRequestedDays(model.getRequestedDays());
        holidayResponse.setYears(model.getYears());

        return holidayResponse;
}

public HolidayResponse toResponse(HolidayRequestEntity entity) {
        HolidayResponse holidayResponse = new HolidayResponse();
        holidayResponse.setId(entity.getId());
        holidayResponse.setEmployeeId(entity.getEmployeeId());
        holidayResponse.setStatus(entity.getStatus());
        holidayResponse.setStartDate(entity.getStartDate().toString())
            ;
        holidayResponse.setEndDate(entity.getEndDate().toString());

        return holidayResponse;
}

public HolidayRequestModel toModel(HolidayRequestEntity entity) {
        HolidayRequestModel model = new HolidayRequestModel();
        model.setId(entity.getId());
        model.setEmployeeId(entity.getEmployeeId());
        model.setRequestStatus(HolidayRequestStatusEnum.valueOf(entity
```

```
                .getStatus()));
            model.setEndDate(entity.getEndDate());
            model.setStartDate(entity.getStartDate());

            return model;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.mapper;

import java.time.LocalDateTime;
import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;

import com.holidaysystem.entity.RequestAlertEntity;
import com.holidaysystem.message.HolidayRequestMessage;
import com.holidaysystem.vo.RequestAlertResponse;

/**
 * Request Alert mapper
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
public class RequestAlertMapper {

        public RequestAlertEntity toEntity(UUID alertId, HolidayRequestMessage
            holidayRequestMessage) {
                RequestAlertEntity entity = new RequestAlertEntity();
        entity.setId(alertId);
        entity.setRequestId(holidayRequestMessage.getId());
        entity.setDate(LocalDateTime.now());
        entity.setCreated(LocalDateTime.now());
        entity.setModified(LocalDateTime.now());

        return entity;
```

```
        }

        public RequestAlertResponse toResponse(RequestAlertEntity entity) {
                RequestAlertResponse requestAlertResponse = new
                    RequestAlertResponse();
                requestAlertResponse.setId(entity.getId());
                requestAlertResponse.setRequestId(entity.getRequestId());
                requestAlertResponse.setDate(entity.getDate().toString());

                return requestAlertResponse;
        }
}
```

] package com.holidaysystem.mapper;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.mapper;

import java.time.LocalDateTime;
import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;

import com.holidaysystem.entity.AccountEntity;
import com.holidaysystem.entity.AuthorizationRoleEntity;
import com.holidaysystem.enumeration.AuthorizationRoleEnum;
import com.holidaysystem.model.AccountDetailsModel;
import com.holidaysystem.vo.AccountResponse;
import com.holidaysystem.vo.RegistrationRequest;

/**
 * Account mapper
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
public class AccountMapper {

        public AccountEntity toEntity(UUID id, RegistrationRequest
```

```
            registrationRequest, String hashedPassWithSalt) {
                AccountEntity account = new AccountEntity();
        account.setId(id);
        account.setEmail(registrationRequest.getEmail());
        account.setPassword(hashedPassWithSalt);
        account.setActive(true);
        account.setCreated(LocalDateTime.now());
                account.setModified(LocalDateTime.now());

        return account;
        }

        public AccountResponse toResponse(AccountDetailsModel model) {
                AccountResponse resp = new AccountResponse();
        resp.setId(model.getId());
        resp.setEmail(model.getEmail());
        resp.setActive(model.getActive());
        resp.setAuthRole(model.getAuthRole().name());

                return resp;
        }

        public AccountDetailsModel toModel(AccountEntity account,
            AuthorizationRoleEntity authRole) {
                AccountDetailsModel accountDetails = new AccountDetailsModel()
                    ;
        accountDetails.setId(account.getId());
        accountDetails.setEmail(account.getEmail());
        accountDetails.setActive(account.getActive());
        accountDetails.setAuthRole(AuthorizationRoleEnum.valueOf(authRole.
            getName()));

        return accountDetails;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.mapper;
```

```
import java.time.LocalDateTime;
import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;

import com.holidaysystem.entity.AccountEntity;
import com.holidaysystem.entity.EmployeeEntity;
import com.holidaysystem.entity.HolidayDetailsEntity;
import com.holidaysystem.enumeration.DepartmentEnum;
import com.holidaysystem.model.EmployeeModel;
import com.holidaysystem.enumeration.EmployeeRoleEnum;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.vo.NewEmployeeRequest;
import com.holidaysystem.vo.EmployeeResponse;

/**
 * Employee mapper
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
public class EmployeeMapper {
        public EmployeeEntity toEntity(NewEmployeeRequest request) {
                EmployeeEntity entity = new EmployeeEntity();
        entity.setId(UUID.randomUUID());
        entity.setAccountId(request.getAccountId());
        entity.setRole(request.getRole());
        entity.setDepartment(request.getDepartment());
        entity.setFirstName(request.getFirstName());
        entity.setLastName(request.getLastName());
        entity.setYears(request.getYears());
        entity.setCreated(LocalDateTime.now());
        entity.setModified(LocalDateTime.now());

        return entity;
        }

        public EmployeeResponse toResponse(EmployeeEntity entity) {
                EmployeeResponse employee = new EmployeeResponse();
                employee.setId(entity.getId());
                employee.setAccountId(entity.getAccountId());
                employee.setFirstName(entity.getFirstName());
                employee.setLastName(entity.getLastName());
                employee.setRole(entity.getRole());
                employee.setDepartment(entity.getDepartment());
                employee.setYears(entity.getYears());

                return employee;
        }

        public EmployeeModel toEmployeeModel(EmployeeEntity entity) {
                EmployeeModel employee = new EmployeeModel();
                employee.setId(entity.getId());
```

```
                employee.setAccountId(entity.getAccountId());
                employee.setFirstName(entity.getFirstName());
                employee.setLastName(entity.getLastName());
                employee.setRole(EmployeeRoleEnum.valueOf(entity.getRole()));
                employee.setDepartment(DepartmentEnum.valueOf(entity.
                    getDepartment()));
                employee.setYears(entity.getYears());

                return employee;
        }

        public EmployeeModel toEmployeeModel(EmployeeEntity employeeEntity,
            AccountEntity accountEntity, HolidayDetailsEntity
            holidayDetailsEntity) {
                EmployeeModel employee = new EmployeeModel();
                employee.setId(employeeEntity.getId());
                employee.setAccountId(employeeEntity.getAccountId());
                employee.setEmail(accountEntity.getEmail());
                employee.setFirstName(employeeEntity.getFirstName());
                employee.setLastName(employeeEntity.getLastName());
                employee.setRole(EmployeeRoleEnum.valueOf(employeeEntity.
                    getRole()));
                employee.setDepartment(DepartmentEnum.valueOf(employeeEntity.
                    getDepartment()));
                employee.setYears(employeeEntity.getYears());
                employee.setTotalDays(holidayDetailsEntity.getTotalDays());
                employee.setTakenDays(holidayDetailsEntity.getTakenDays());

                return employee;
        }

        public EmployeeResponse toResponse(EmployeeModel model) {
                EmployeeResponse employee = new EmployeeResponse();
                employee.setId(model.getId());
                employee.setAccountId(model.getAccountId());
                employee.setEmail(model.getEmail());
                employee.setFirstName(model.getFirstName());
                employee.setLastName(model.getLastName());
                employee.setRole(model.getRole().name());
                employee.setDepartment(model.getDepartment().name());
                employee.setYears(model.getYears());
                employee.setTotalDays(model.getTotalDays());
                employee.setTakenDays(model.getTakenDays());
                employee.setHolidayStatus(model.getHolidayStatus() == null?
                    HolidayStatusEnum.ON_DUTY.name()
                                : model.getHolidayStatus().name());

                return employee;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
```

```
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.mapper;

import java.time.LocalDateTime;
import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;

import com.fasterxml.jackson.core.JsonGenerationException;
import com.fasterxml.jackson.core.JsonProcessingException;
import com.fasterxml.jackson.databind.JsonMappingException;
import com.fasterxml.jackson.databind.ObjectMapper;
import com.fasterxml.jackson.databind.ObjectWriter;
import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.HolidayRequestEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.message.HolidayRequestMessage;
import com.holidaysystem.model.HolidayRequestModel;
import com.holidaysystem.vo.HolidayRequest;
import com.holidaysystem.vo.HolidayResponse;

/**
 * Holiday Request mapper
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
public class HolidayRequestMapper {
        public HolidayRequestEntity toEntity(UUID id, HolidayRequest request)
            {
                HolidayRequestEntity entity = new HolidayRequestEntity();
        entity.setId(id);
        entity.setEmployeeId(request.getEmployeeId());
        entity.setStatus(request.getStatus());
        entity.setStartDate(LocalDateTime.parse(request.getStartDate(),
            DateUtils.FORMATTER));
        entity.setEndDate(LocalDateTime.parse(request.getEndDate(), DateUtils.
            FORMATTER));
        entity.setCreated(LocalDateTime.now());
        entity.setModified(LocalDateTime.now());
```

```
return entity;
}

public HolidayRequestEntity toEntity(HolidayRequestMessage
    holidayRequestMessage) {
        HolidayRequestEntity entity = new HolidayRequestEntity();
entity.setId(holidayRequestMessage.getId());
entity.setEmployeeId(holidayRequestMessage.getEmployeeId());
entity.setStatus(holidayRequestMessage.getStatus());
entity.setStartDate(LocalDateTime.parse(holidayRequestMessage.
    getStartDate(), DateUtils.FORMATTER));
entity.setEndDate(LocalDateTime.parse(holidayRequestMessage.getEndDate
    (), DateUtils.FORMATTER));
entity.setCreated(LocalDateTime.now());
entity.setModified(LocalDateTime.now());

return entity;
}

public HolidayRequestMessage toMessage(UUID id, HolidayRequest request
    ) {
        HolidayRequestMessage message = new HolidayRequestMessage();
        message.setId(id);
        message.setEmployeeId(request.getEmployeeId());
        message.setStatus(request.getStatus());
        message.setStartDate(request.getStartDate());
        message.setEndDate(request.getEndDate());

return message;
}

public HolidayRequestMessage toMessage(String jsonMessage) {

try {
        ObjectMapper objectMapper = new ObjectMapper();
        HolidayRequestMessage message = objectMapper.readValue(
            jsonMessage, HolidayRequestMessage.class);

    return message;
}
catch (JsonGenerationException | JsonMappingException  e) {
    e.printStackTrace();
}
catch(JsonProcessingException e) {
        e.printStackTrace();
}
catch(Exception e) {
        e.printStackTrace();
}

return null;
}

public String toJson(UUID id, HolidayRequest request) {
```

```
        HolidayRequestMessage message = toMessage(id, request);


try {
        ObjectWriter ow = new ObjectMapper().writer().
            withDefaultPrettyPrinter();
        String json = ow.writeValueAsString(message);

    return json;
}
catch (JsonGenerationException | JsonMappingException  e) {
    e.printStackTrace();
}
catch(JsonProcessingException e) {
        e.printStackTrace();
}
catch(Exception e) {
        e.printStackTrace();
}

return null;
}

public HolidayRequestEntity toEntity(HolidayRequest updated,
    HolidayRequestEntity db) {
        HolidayRequestEntity entity = new HolidayRequestEntity();
entity.setId(db.getId());
entity.setEmployeeId(updated.getEmployeeId());
entity.setStatus(updated.getStatus());
entity.setStartDate(LocalDateTime.parse(updated.getStartDate(),
    DateUtils.FORMATTER));
entity.setEndDate(LocalDateTime.parse(updated.getEndDate(), DateUtils.
   FORMATTER));
entity.setCreated(db.getCreated());
entity.setModified(LocalDateTime.now());

return entity;
}

public HolidayResponse toResponse(HolidayRequestModel model) {
        HolidayResponse holidayResponse = new HolidayResponse();
        holidayResponse.setId(model.getId());
        holidayResponse.setEmployeeId(model.getEmployeeId());
        holidayResponse.setStatus(model.getRequestStatus().name());
        holidayResponse.setStartDate(model.getStartDate().toString());
        holidayResponse.setEndDate(model.getEndDate().toString());
        holidayResponse.setTakenDays(model.getTakenDays());
        holidayResponse.setTotalDays(model.getTotalDays());
        holidayResponse.setRequestedDays(model.getRequestedDays());
        holidayResponse.setYears(model.getYears());

        return holidayResponse;
}
```

```
        public HolidayResponse toResponse(HolidayRequestEntity entity) {
                HolidayResponse holidayResponse = new HolidayResponse();
                holidayResponse.setId(entity.getId());
                holidayResponse.setEmployeeId(entity.getEmployeeId());
                holidayResponse.setStatus(entity.getStatus());
                holidayResponse.setStartDate(entity.getStartDate().toString())
                    ;
                holidayResponse.setEndDate(entity.getEndDate().toString());

                return holidayResponse;
        }

        public HolidayRequestModel toModel(HolidayRequestEntity entity) {
                HolidayRequestModel model = new HolidayRequestModel();
                model.setId(entity.getId());
                model.setEmployeeId(entity.getEmployeeId());
                model.setRequestStatus(HolidayRequestStatusEnum.valueOf(entity
                    .getStatus()));
                model.setEndDate(entity.getEndDate());
                model.setStartDate(entity.getStartDate());

                return model;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.mapper;

import java.time.LocalDateTime;
import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;

import com.holidaysystem.entity.RequestAlertEntity;
import com.holidaysystem.message.HolidayRequestMessage;
import com.holidaysystem.vo.RequestAlertResponse;

/**
```

```
 * Request Alert mapper
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
public class RequestAlertMapper {

        public RequestAlertEntity toEntity(UUID alertId, HolidayRequestMessage
             holidayRequestMessage) {
                RequestAlertEntity entity = new RequestAlertEntity();
        entity.setId(alertId);
        entity.setRequestId(holidayRequestMessage.getId());
        entity.setDate(LocalDateTime.now());
        entity.setCreated(LocalDateTime.now());
        entity.setModified(LocalDateTime.now());

        return entity;
        }

        public RequestAlertResponse toResponse(RequestAlertEntity entity) {
                RequestAlertResponse requestAlertResponse = new
                    RequestAlertResponse();
                requestAlertResponse.setId(entity.getId());
                requestAlertResponse.setRequestId(entity.getRequestId());
                requestAlertResponse.setDate(entity.getDate().toString());

                return requestAlertResponse;
        }
}



/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.message;

import java.util.UUID;

/**
 * Model of the message in message system
```

```
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public class HolidayRequestMessage {
        private static final long serialVersionUID = 1L;

    private UUID id;
    private UUID employeeId;
        private String startDate;
        private String endDate;
        private String status;

    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setEmployeeId(UUID employeeId) {this.employeeId = employeeId;
       }
    public UUID getEmployeeId() {return this.employeeId;}
    public void setStatus(String status) {this.status = status; }
    public String getStatus() {return this.status;}
    public void setStartDate(String startDate) {this.startDate = startDate; }
    public String getStartDate() {return this.startDate;}
    public void setEndDate(String endDate) {this.endDate = endDate; }
    public String getEndDate() {return this.endDate;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.message;

import java.util.UUID;

import javax.ejb.ActivationConfigProperty;
import javax.ejb.MessageDriven;
import javax.enterprise.context.ApplicationScoped;
import javax.inject.Inject;
import javax.jms.JMSException;
import javax.jms.MapMessage;
import javax.jms.Message;
import javax.jms.MessageListener;
```

```
import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.entity.RequestAlertEntity;
import com.holidaysystem.mail.MailService;
import com.holidaysystem.mapper.HolidayRequestMapper;
import com.holidaysystem.mapper.RequestAlertMapper;
import com.holidaysystem.repository.RequestAlertRepository;

/**
 * HolidayRequestMessageBean listener of the new messages in message queue
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@MessageDriven(
                activationConfig = { @ActivationConfigProperty(
                                propertyName = "destination",
                                propertyValue = Constants.JMS_MESSAGE_QUEUE),
                                @ActivationConfigProperty(propertyName = "
                                    destinationType",
                                propertyValue = "javax.jms.Queue")
                },
                mappedName = Constants.JMS_MESSAGE_QUEUE)
public class HolidayRequestMessageBean implements MessageListener {

        private static final Logger logger = Logger.getLogger(
           HolidayRequestMessageBean.class);

        @Inject
    private RequestAlertRepository requestAlertRepository;
        @Inject
        private HolidayRequestMapper holidayRequestMapper;
        @Inject
        private RequestAlertMapper requestAlertMapper;
        @Inject
        private MailService mailService;

    public void onMessage(Message message) {
        logger.info("Message received from message queue");

        try {
                MapMessage requestMsg = (MapMessage) message;
                        String jsonRequest = requestMsg.getString(Constants.
                           QUEUE_KEY_MESSAGE);
                        logger.info(String.format("The jsonRequest: %s ",
                           jsonRequest));

                        HolidayRequestMessage holidayRequestMessage =
                           holidayRequestMapper.toMessage(jsonRequest);
                        RequestAlertEntity entity = requestAlertMapper.
                           toEntity(UUID.randomUUID(), holidayRequestMessage);

                        requestAlertRepository.save(entity);
```

```
                        mailService.send();

                } catch (JMSException e) {
                        logger.error(e.getMessage(), e);
                }catch (Exception e) {
                        logger.error(e.getMessage(), e);
                }
    }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.message;

import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;
import javax.inject.Inject;
import javax.jms.Connection;
import javax.jms.ConnectionFactory;
import javax.jms.JMSException;
import javax.jms.MapMessage;
import javax.jms.MessageProducer;
import javax.jms.Queue;
import javax.jms.Session;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.mapper.HolidayRequestMapper;
import com.holidaysystem.vo.HolidayRequest;

/**
 * Sends a message to message system
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
```

```
 *
 */
@ApplicationScoped
public class HolidayRequestMQProducer {

        private static final Logger logger = Logger.getLogger(
            HolidayRequestMQProducer.class);

        @Inject
    private HolidayRequestMapper holidayRequestMapper;

        public void publish(UUID id, HolidayRequest request) {

        String jsonEntity = holidayRequestMapper.toJson(id, request);

                try {
                        Context jndiContext = new InitialContext();
                        ConnectionFactory factory = (ConnectionFactory)
                            jndiContext.lookup(Constants.
                            JNDI_CONNECTION_FACTORY);
                        Queue calculationQueue = (Queue) jndiContext.lookup(
                            Constants.JMS_MESSAGE_QUEUE);
                        Connection connect = factory.createConnection();

                        Session session = connect.createSession(false, Session
                            .AUTO_ACKNOWLEDGE);

                        MessageProducer sender = session.createProducer(
                            calculationQueue);
                        MapMessage message = session.createMapMessage();
                        message.setString(Constants.QUEUE_KEY_MESSAGE,
                            jsonEntity);

                        logger.debug("Sending message");

                        sender.send(message);
                        connect.close();

                } catch (NamingException e) {
                        logger.error(e.getMessage(), e);
                } catch (JMSException e) {
                        logger.error(e.getMessage(), e);
                }catch (Exception e) {
                        logger.error(e.getMessage(), e);
                }
        }
}
```

] package com.holidaysystem.message;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
```

```
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.message;

import java.util.UUID;

/**
 * Model of the message in message system
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public class HolidayRequestMessage {
        private static final long serialVersionUID = 1L;

    private UUID id;
    private UUID employeeId;
        private String startDate;
        private String endDate;
        private String status;

    public void setId(UUID id) {this.id = id; }
    public UUID getId() {return this.id;}
    public void setEmployeeId(UUID employeeId) {this.employeeId = employeeId;
       }
    public UUID getEmployeeId() {return this.employeeId;}
    public void setStatus(String status) {this.status = status; }
    public String getStatus() {return this.status;}
    public void setStartDate(String startDate) {this.startDate = startDate; }
    public String getStartDate() {return this.startDate;}
    public void setEndDate(String endDate) {this.endDate = endDate; }
    public String getEndDate() {return this.endDate;}
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
```

```
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.message;

import java.util.UUID;

import javax.ejb.ActivationConfigProperty;
import javax.ejb.MessageDriven;
import javax.enterprise.context.ApplicationScoped;
import javax.inject.Inject;
import javax.jms.JMSException;
import javax.jms.MapMessage;
import javax.jms.Message;
import javax.jms.MessageListener;
import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.entity.RequestAlertEntity;
import com.holidaysystem.mail.MailService;
import com.holidaysystem.mapper.HolidayRequestMapper;
import com.holidaysystem.mapper.RequestAlertMapper;
import com.holidaysystem.repository.RequestAlertRepository;

/**
 * HolidayRequestMessageBean listener of the new messages in message queue
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@MessageDriven(
                activationConfig = { @ActivationConfigProperty(
                                propertyName = "destination",
                                propertyValue = Constants.JMS_MESSAGE_QUEUE),
                                @ActivationConfigProperty(propertyName = "
                                    destinationType",
                                propertyValue = "javax.jms.Queue")
                },
                mappedName = Constants.JMS_MESSAGE_QUEUE)
public class HolidayRequestMessageBean implements MessageListener {

        private static final Logger logger = Logger.getLogger(
            HolidayRequestMessageBean.class);

        @Inject
    private RequestAlertRepository requestAlertRepository;
        @Inject
        private HolidayRequestMapper holidayRequestMapper;
        @Inject
        private RequestAlertMapper requestAlertMapper;
        @Inject
```

```
        private MailService mailService;

    public void onMessage(Message message) {
        logger.info("Message received from message queue");

        try {
                MapMessage requestMsg = (MapMessage) message;
                        String jsonRequest = requestMsg.getString(Constants.
                            QUEUE_KEY_MESSAGE);
                        logger.info(String.format("The jsonRequest: %s ",
                            jsonRequest));

                        HolidayRequestMessage holidayRequestMessage =
                            holidayRequestMapper.toMessage(jsonRequest);
                        RequestAlertEntity entity = requestAlertMapper.
                            toEntity(UUID.randomUUID(), holidayRequestMessage);

                        requestAlertRepository.save(entity);

                        mailService.send();

                } catch (JMSException e) {
                        logger.error(e.getMessage(), e);
                }catch (Exception e) {
                        logger.error(e.getMessage(), e);
                }
    }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.message;

import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;
import javax.inject.Inject;
import javax.jms.Connection;
import javax.jms.ConnectionFactory;
```

```
import javax.jms.JMSException;
import javax.jms.MapMessage;
import javax.jms.MessageProducer;
import javax.jms.Queue;
import javax.jms.Session;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.mapper.HolidayRequestMapper;
import com.holidaysystem.vo.HolidayRequest;

/**
 * Sends a message to message system
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
public class HolidayRequestMQProducer {

        private static final Logger logger = Logger.getLogger(
            HolidayRequestMQProducer.class);

        @Inject
    private HolidayRequestMapper holidayRequestMapper;

        public void publish(UUID id, HolidayRequest request) {

        String jsonEntity = holidayRequestMapper.toJson(id, request);

                try {
                        Context jndiContext = new InitialContext();
                        ConnectionFactory factory = (ConnectionFactory)
                            jndiContext.lookup(Constants.
                            JNDI_CONNECTION_FACTORY);
                        Queue calculationQueue = (Queue) jndiContext.lookup(
                            Constants.JMS_MESSAGE_QUEUE);
                        Connection connect = factory.createConnection();

                        Session session = connect.createSession(false, Session
                            .AUTO_ACKNOWLEDGE);

                        MessageProducer sender = session.createProducer(
                            calculationQueue);
                        MapMessage message = session.createMapMessage();
                        message.setString(Constants.QUEUE_KEY_MESSAGE,
                            jsonEntity);

                        logger.debug("Sending message");

                        sender.send(message);
```

```
                        connect.close();

                } catch (NamingException e) {
                        logger.error(e.getMessage(), e);
                } catch (JMSException e) {
                        logger.error(e.getMessage(), e);
                }catch (Exception e) {
                        logger.error(e.getMessage(), e);
                }
        }
}


/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.model;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;

import com.holidaysystem.enumeration.AuthorizationRoleEnum;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Account details model
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class AccountDetailsModel implements Serializable {
        private static final long serialVersionUID = 1L;
```

```
        @NotBlank
    private UUID id;

        @NotBlank
    private String email;

        @NotNull
    private Boolean active;

        @NotNull
    private AuthorizationRoleEnum authRole;

    public UUID getId() {
        return this.id;
    }
    public void setId(UUID id) {
        this.id = id;
    }
    public String getEmail() {
        return this.email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public Boolean getActive() {
        return this.active;
    }
    public void setActive(Boolean active) {
        this.active = active;
    }
    public AuthorizationRoleEnum getAuthRole() {
        return this.authRole;
    }
    public void setAuthRole(AuthorizationRoleEnum authRole) {
        this.authRole = authRole;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
```

```
 */
package com.holidaysystem.model;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.time.LocalDateTime;

import javax.validation.constraints.NotNull;

/**
 * Alternative date model keeps left and right border of holiday period
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class AlternativeDateModel implements Serializable {
        private static final long serialVersionUID = 1L;

        @NotNull
        private LocalDateTime startDate;
        @NotNull
        private LocalDateTime endDate;

    public LocalDateTime getStartDate() {
        return this.startDate;
    }
    public void setStartDate(LocalDateTime startDate) {
        this.startDate = startDate;
    }
    public void setEndDate(LocalDateTime endDate) {
        this.endDate = endDate;
    }
    public LocalDateTime getEndDate() {
        return this.endDate;
    }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
```

```
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.model;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.List;
import java.util.UUID;

/**
 * Alternative dates model includes data from different entities
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class AlternativeDatesModel implements Serializable {
        private static final long serialVersionUID = 1L;

        @NotBlank
    private UUID requestId;

    @NotNull
    private List<AlternativeDateModel> alternativeDates;

    public UUID getRequestId() {
        return this.requestId;
    }
    public void setRequestId(UUID requestId) {
        this.requestId = requestId;
    }
    public List<AlternativeDateModel> getAlternativeDates() {
        return this.alternativeDates;
    }
    public void setId(List<AlternativeDateModel> alternativeDates) {
        this.alternativeDates.clear();
        this.alternativeDates.addAll(alternativeDates);
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
```

```
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.model;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.PositiveOrZero;

import com.holidaysystem.enumeration.DepartmentEnum;
import com.holidaysystem.enumeration.EmployeeRoleEnum;
import com.holidaysystem.enumeration.HolidayStatusEnum;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Employee model includes data from different entities
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class EmployeeModel implements Serializable {
        private static final long serialVersionUID = 1L;

        @NotBlank
    private UUID id;

    private String email;

    @NotBlank
    private String firstName;

    @NotBlank
    private String lastName;

    @NotBlank
    private UUID accountId;

    @NotBlank
    private EmployeeRoleEnum role;
```

```
@NotBlank
private DepartmentEnum department;

@PositiveOrZero
private Integer years;

@PositiveOrZero
private Integer totalDays;

@PositiveOrZero
private Integer takenDays;

@NotNull
private HolidayStatusEnum holidayStatus;

public UUID getId() {
    return this.id;
}
public void setId(UUID id) {
    this.id = id;
}
public String getFirstName() {
    return this.firstName;
}
public void setFirstName(String firstName) {
    this.firstName = firstName;
}
public String getLastName() {
    return this.lastName;
}
public void setEmail(String email) {
    this.email = email;
}
public String getEmail() {
    return this.email;
}
public void setLastName(String lastName) {
    this.lastName = lastName;
}
public UUID getAccountId() {
    return this.accountId;
}
public void setAccountId(UUID accountId) {
    this.accountId = accountId;
}
public EmployeeRoleEnum getRole() {
    return this.role;
}
public void setRole(EmployeeRoleEnum role) {
    this.role = role;
}
public DepartmentEnum getDepartment() {
    return this.department;
```

```
    }
    public void setDepartment(DepartmentEnum department) {
        this.department = department;
    }
    public void setYears(Integer years) {this.years = years; }
    public Integer getYears() {return this.years;}
    public void setTotalDays(Integer totalDays) {this.totalDays = totalDays; }
    public Integer getTotalDays() {return this.totalDays;}
    public void setTakenDays(Integer takenDays) {this.takenDays = takenDays; }
    public Integer getTakenDays() {return this.takenDays;}
    public void setHolidayStatus(HolidayStatusEnum holidayStatus) {this.
        holidayStatus = holidayStatus; }
    public HolidayStatusEnum getHolidayStatus() {return this.holidayStatus;}
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.model;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.PositiveOrZero;

import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.enumeration.HolidayStatusEnum;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.time.LocalDateTime;
import java.util.UUID;

/**
 * Holiday Request model includes data from different entities
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
```

```
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class HolidayRequestModel implements Serializable {
        private static final long serialVersionUID = 1L;

        @NotBlank
    private UUID id;

    @NotBlank
    private UUID employeeId;

    @NotNull
    private HolidayStatusEnum holidayStatus;

    @NotNull
    private HolidayRequestStatusEnum requestStatus;

    @PositiveOrZero
    private Integer requestedDays;

    @NotNull
    private LocalDateTime startDate;

    @NotNull
    private LocalDateTime endDate;

    @PositiveOrZero
    private Integer years;

    @PositiveOrZero
    private Integer totalDays;

    @PositiveOrZero
    private Integer takenDays;

    public UUID getId() {
        return this.id;
    }
    public void setId(UUID id) {
        this.id = id;
    }
    public UUID getEmployeeId() {
        return this.employeeId;
    }
    public void setEmployeeId(UUID employeeId) {
        this.employeeId = employeeId;
    }
    public HolidayStatusEnum getHolidayStatus() {
        return this.holidayStatus;
    }
    public void setHolidayStatus(HolidayStatusEnum holidayStatus) {
        this.holidayStatus = holidayStatus;
    }
    public HolidayRequestStatusEnum getRequestStatus() {
```

```
        return this.requestStatus;
    }
    public void setRequestStatus(HolidayRequestStatusEnum requestStatus) {
        this.requestStatus = requestStatus;
    }
    public LocalDateTime getStartDate() {
        return this.startDate;
    }
    public void setStartDate(LocalDateTime startDate) {
        this.startDate = startDate;
    }
    public void setEndDate(LocalDateTime endDate) {
        this.endDate = endDate;
    }
    public LocalDateTime getEndDate() {
        return this.endDate;
    }
    public void setYears(Integer years) {this.years = years; }
    public Integer getYears() {return this.years;}
    public void setTotalDays(Integer totalDays) {this.totalDays = totalDays; }
    public Integer getTotalDays() {return this.totalDays;}
    public void setTakenDays(Integer takenDays) {this.takenDays = takenDays; }
    public Integer getTakenDays() {return this.takenDays;}
    public void setRequestedDays(Integer requestedDays) {this.requestedDays =
       requestedDays; }
    public Integer getRequestedDays() {return this.requestedDays;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.holidaysystem.model;

/**
 * Pair class contains left and right included border values of the range
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 * @param <L>
 * @param <R>
```

```
 */
public class Pair<L, R> {
        private L leftBorder;
        private R rightBorder;

        public Pair(L leftBorder, R rightBorder) {
                this.leftBorder = leftBorder;
                this.rightBorder = rightBorder;
        }

        public L getLeftBorder() {return this.leftBorder;}
        public void setLeftBorder(L leftBorder) {this.leftBorder = leftBorder
            ;}
        public R getRigthBorder() {return this.rightBorder;}
        public void setRightBorder(R rightBorder) {this.rightBorder =
            rightBorder;}
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.model;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.PositiveOrZero;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Prioritized  request model for holiday request
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
```

```
public class PrioritizedRequestModel implements Serializable {

    @NotBlank
    private UUID requestId;

    @PositiveOrZero
    private Integer takenDays;

    @PositiveOrZero
    private Integer requestedDays;

    public UUID getRequestId() {
        return this.requestId;
    }
    public void setRequestId(UUID requestId) {
        this.requestId = requestId;
    }
    public void setTakenDays(Integer takenDays) {this.takenDays = takenDays; }
    public Integer getTakenDays() {return this.takenDays;}
    public void setRequestedDays(Integer requestedDays) {this.requestedDays =
       requestedDays; }
    public Integer getRequestedDays() {return this.requestedDays;}

}
```

] package com.holidaysystem.model;

```
/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.model;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;

import com.holidaysystem.enumeration.AuthorizationRoleEnum;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;
```

```
import java.io.Serializable;
import java.util.UUID;

/**
 * Account details model
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class AccountDetailsModel implements Serializable {
        private static final long serialVersionUID = 1L;

        @NotBlank
    private UUID id;

        @NotBlank
    private String email;

        @NotNull
    private Boolean active;

        @NotNull
    private AuthorizationRoleEnum authRole;

    public UUID getId() {
        return this.id;
    }
    public void setId(UUID id) {
        this.id = id;
    }
    public String getEmail() {
        return this.email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public Boolean getActive() {
        return this.active;
    }
    public void setActive(Boolean active) {
        this.active = active;
    }
    public AuthorizationRoleEnum getAuthRole() {
        return this.authRole;
    }
    public void setAuthRole(AuthorizationRoleEnum authRole) {
        this.authRole = authRole;
    }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
```

```
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.model;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.time.LocalDateTime;

import javax.validation.constraints.NotNull;

/**
 * Alternative date model keeps left and right border of holiday period
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class AlternativeDateModel implements Serializable {
        private static final long serialVersionUID = 1L;

        @NotNull
        private LocalDateTime startDate;
        @NotNull
        private LocalDateTime endDate;

    public LocalDateTime getStartDate() {
        return this.startDate;
    }
    public void setStartDate(LocalDateTime startDate) {
        this.startDate = startDate;
    }
    public void setEndDate(LocalDateTime endDate) {
        this.endDate = endDate;
    }
    public LocalDateTime getEndDate() {
        return this.endDate;
    }
}
```

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.model;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.List;
import java.util.UUID;

/**
 * Alternative dates model includes data from different entities
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class AlternativeDatesModel implements Serializable {
        private static final long serialVersionUID = 1L;

        @NotBlank
    private UUID requestId;

    @NotNull
    private List<AlternativeDateModel> alternativeDates;

    public UUID getRequestId() {
        return this.requestId;
    }
    public void setRequestId(UUID requestId) {
        this.requestId = requestId;
    }
    public List<AlternativeDateModel> getAlternativeDates() {
        return this.alternativeDates;
```

```
    }
    public void setId(List<AlternativeDateModel> alternativeDates) {
        this.alternativeDates.clear();
        this.alternativeDates.addAll(alternativeDates);
    }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.model;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.PositiveOrZero;

import com.holidaysystem.enumeration.DepartmentEnum;
import com.holidaysystem.enumeration.EmployeeRoleEnum;
import com.holidaysystem.enumeration.HolidayStatusEnum;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Employee model includes data from different entities
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class EmployeeModel implements Serializable {
        private static final long serialVersionUID = 1L;

        @NotBlank
    private UUID id;
```

```
private String email;

@NotBlank
private String firstName;

@NotBlank
private String lastName;

@NotBlank
private UUID accountId;

@NotBlank
private EmployeeRoleEnum role;

@NotBlank
private DepartmentEnum department;

@PositiveOrZero
private Integer years;

@PositiveOrZero
private Integer totalDays;

@PositiveOrZero
private Integer takenDays;

@NotNull
private HolidayStatusEnum holidayStatus;

public UUID getId() {
    return this.id;
}
public void setId(UUID id) {
    this.id = id;
}
public String getFirstName() {
    return this.firstName;
}
public void setFirstName(String firstName) {
    this.firstName = firstName;
}
public String getLastName() {
    return this.lastName;
}
public void setEmail(String email) {
    this.email = email;
}
public String getEmail() {
    return this.email;
}
public void setLastName(String lastName) {
    this.lastName = lastName;
}
public UUID getAccountId() {
```

```
        return this.accountId;
    }
    public void setAccountId(UUID accountId) {
        this.accountId = accountId;
    }
    public EmployeeRoleEnum getRole() {
        return this.role;
    }
    public void setRole(EmployeeRoleEnum role) {
        this.role = role;
    }
    public DepartmentEnum getDepartment() {
        return this.department;
    }
    public void setDepartment(DepartmentEnum department) {
        this.department = department;
    }
    public void setYears(Integer years) {this.years = years; }
    public Integer getYears() {return this.years;}
    public void setTotalDays(Integer totalDays) {this.totalDays = totalDays; }
    public Integer getTotalDays() {return this.totalDays;}
    public void setTakenDays(Integer takenDays) {this.takenDays = takenDays; }
    public Integer getTakenDays() {return this.takenDays;}
    public void setHolidayStatus(HolidayStatusEnum holidayStatus) {this.
        holidayStatus = holidayStatus; }
    public HolidayStatusEnum getHolidayStatus() {return this.holidayStatus;}
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.model;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.PositiveOrZero;

import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.enumeration.HolidayStatusEnum;
```

```
import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.time.LocalDateTime;
import java.util.UUID;

/**
 * Holiday Request model includes data from different entities
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class HolidayRequestModel implements Serializable {
        private static final long serialVersionUID = 1L;

        @NotBlank
    private UUID id;

    @NotBlank
    private UUID employeeId;

    @NotNull
    private HolidayStatusEnum holidayStatus;

    @NotNull
    private HolidayRequestStatusEnum requestStatus;

    @PositiveOrZero
    private Integer requestedDays;

    @NotNull
    private LocalDateTime startDate;

    @NotNull
    private LocalDateTime endDate;

    @PositiveOrZero
    private Integer years;

    @PositiveOrZero
    private Integer totalDays;

    @PositiveOrZero
    private Integer takenDays;

    public UUID getId() {
        return this.id;
    }
    public void setId(UUID id) {
        this.id = id;
    }
```

```
    public UUID getEmployeeId() {
        return this.employeeId;
    }
    public void setEmployeeId(UUID employeeId) {
        this.employeeId = employeeId;
    }
    public HolidayStatusEnum getHolidayStatus() {
        return this.holidayStatus;
    }
    public void setHolidayStatus(HolidayStatusEnum holidayStatus) {
        this.holidayStatus = holidayStatus;
    }
    public HolidayRequestStatusEnum getRequestStatus() {
        return this.requestStatus;
    }
    public void setRequestStatus(HolidayRequestStatusEnum requestStatus) {
        this.requestStatus = requestStatus;
    }
    public LocalDateTime getStartDate() {
        return this.startDate;
    }
    public void setStartDate(LocalDateTime startDate) {
        this.startDate = startDate;
    }
    public void setEndDate(LocalDateTime endDate) {
        this.endDate = endDate;
    }
    public LocalDateTime getEndDate() {
        return this.endDate;
    }
    public void setYears(Integer years) {this.years = years; }
    public Integer getYears() {return this.years;}
    public void setTotalDays(Integer totalDays) {this.totalDays = totalDays; }
    public Integer getTotalDays() {return this.totalDays;}
    public void setTakenDays(Integer takenDays) {this.takenDays = takenDays; }
    public Integer getTakenDays() {return this.takenDays;}
    public void setRequestedDays(Integer requestedDays) {this.requestedDays =
       requestedDays; }
    public Integer getRequestedDays() {return this.requestedDays;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
```

```
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.holidaysystem.model;

/**
 * Pair class contains left and right included border values of the range
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 * @param <L>
 * @param <R>
 */
public class Pair<L, R> {
        private L leftBorder;
        private R rightBorder;

        public Pair(L leftBorder, R rightBorder) {
                this.leftBorder = leftBorder;
                this.rightBorder = rightBorder;
        }

        public L getLeftBorder() {return this.leftBorder;}
        public void setLeftBorder(L leftBorder) {this.leftBorder = leftBorder
            ;}
        public R getRigthBorder() {return this.rightBorder;}
        public void setRightBorder(R rightBorder) {this.rightBorder =
            rightBorder;}
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.model;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.PositiveOrZero;

import jakarta.xml.bind.annotation.XmlAccessType;
```

```
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Prioritized  request model for holiday request
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class PrioritizedRequestModel implements Serializable {

    @NotBlank
    private UUID requestId;

    @PositiveOrZero
    private Integer takenDays;

    @PositiveOrZero
    private Integer requestedDays;

    public UUID getRequestId() {
        return this.requestId;
    }
    public void setRequestId(UUID requestId) {
        this.requestId = requestId;
    }
    public void setTakenDays(Integer takenDays) {this.takenDays = takenDays; }
    public Integer getTakenDays() {return this.takenDays;}
    public void setRequestedDays(Integer requestedDays) {this.requestedDays =
       requestedDays; }
    public Integer getRequestedDays() {return this.requestedDays;}

}



/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
```

```
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;

import javax.annotation.Resource;
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.sql.DataSource;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.AccountEntity;

import jakarta.inject.Named;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.time.LocalDateTime;

/**
 * Account Repository implementation using java.sql.PreparedStatement
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 */
@ApplicationScoped
@Default
public class AccountRepository implements IAccountRepository {

        private static final Logger logger = Logger.getLogger(
           AccountRepository.class);

        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    DataSource dataSource;

        @Override
        public AccountEntity findById(UUID accountId) {
                try (Connection connection = dataSource.getConnection()) {
                        final String query = "SELECT acc.id, acc.email, acc.
                           password, acc.active, "
                                        + "acc.auth_roleid, acc.created, acc.
                                           modified "
                                        + "FROM account acc "
                                        + "WHERE acc.id = ?";

                        try (PreparedStatement stmt = connection.
                           prepareStatement(query)) {
                                stmt.setObject(1, accountId);

                                ResultSet rs = stmt.executeQuery();
```

```
                        while(rs.next()) {
                                AccountEntity account = new
                                    AccountEntity();
                                account.setId(UUID.fromString(rs.
                                    getString("id")));
                                account.setPassword(rs.getString("
                                    password"));
                                account.setEmail(rs.getString("email")
                                    );
                                account.setActive(rs.getBoolean("
                                    active"));
                                account.setAuthRoleId(UUID.fromString(
                                    rs.getString("auth_roleid")));
                                account.setCreated(LocalDateTime.parse
                                    (rs.getString("created"), DateUtils
                                    .FORMATTER));
                                account.setModified(LocalDateTime.
                                    parse(rs.getString("modified"),
                                    DateUtils.FORMATTER));

                                return account;
                        }

                        return null;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public AccountEntity findByEmail(String email) {

        try (Connection connection = dataSource.getConnection()) {

                String query = "SELECT acc.id, acc.email, acc.password
                    , acc.active, acc.auth_roleid, acc.created, acc.
                    modified "
                                + "FROM account acc "
                                + "WHERE acc.email = ?";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setString(1, email);

                        ResultSet rs = stmt.executeQuery();
                        while(rs.next()) {
                                AccountEntity account = new
                                    AccountEntity();
                                account.setId(UUID.fromString(rs.
                                    getString("id")));
                                account.setPassword(rs.getString("
```

```
                                        password"));
                                account.setEmail(rs.getString("email")
                                    );
                                account.setActive(rs.getBoolean("
                                    active"));
                                account.setAuthRoleId(UUID.fromString(
                                    rs.getString("auth_roleid")));
                                account.setCreated(LocalDateTime.parse
                                    (rs.getString("created"), DateUtils
                                    .FORMATTER));
                                account.setModified(LocalDateTime.
                                    parse(rs.getString("modified"),
                                    DateUtils.FORMATTER));

                                return account;
                        }

                        return null;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public AccountEntity findByEmailAndPassword(String email, String
    password) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT acc.id, acc.email, acc.
                    password, acc.active, acc.auth_roleid, acc.created,
                     acc.modified "
                                + "FROM account acc "
                                + "WHERE acc.email = ? AND "
                                + "crypt(?, acc.password) = acc.
                                    password";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setString(1, email);
                        stmt.setString(2, password);

                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                AccountEntity account = new
                                    AccountEntity();
                                account.setId(UUID.fromString(rs.
                                    getString("id")));
                                account.setPassword(rs.getString("
                                    password"));
                                account.setEmail(rs.getString("email")
```

```
                                        );
                                account.setActive(rs.getBoolean("
                                    active"));
                                account.setAuthRoleId(UUID.fromString(
                                    rs.getString("auth_roleid")));
                                account.setCreated(LocalDateTime.parse
                                    (rs.getString("created"), DateUtils
                                    .FORMATTER));
                                account.setModified(LocalDateTime.
                                    parse(rs.getString("modified"),
                                    DateUtils.FORMATTER));

                                return account;
                        }
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
    }

    @Override
    public boolean save(AccountEntity account) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "INSERT INTO account (id, email,
                    password, active, auth_roleid, created, modified) "
                                + "VALUES (?,?,?,?,?,?,?);";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, account.getId());
                        stmt.setString(2, account.getEmail());
                        stmt.setString(3, account.getPassword());
                        stmt.setObject(4, account.getActive());
                        stmt.setObject(5, account.getAuthRoleId());
                        stmt.setObject(6, account.getCreated());
                        stmt.setObject(7, account.getModified());

                        if (stmt.executeUpdate() == 1) {
                            return true;
                        } else {
                            return false;
                        }
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return false;
    }

public String generateHashedPassword(String password) {
```

```
                try (Connection connection = dataSource.getConnection()) {

                        final String query = "SELECT crypt(?, gen_salt('bf',
                            8))";

                        try (PreparedStatement stmt = connection.
                            prepareStatement(query)) {
                                stmt.setObject(1, password);

                                ResultSet rs = stmt.executeQuery();

                                String hashedPass = "";

                                while(rs.next()) {
                                        hashedPass = rs.getString(1);
                                }

                                return hashedPass;
                        }
                } catch(Exception ex) {
                        logger.error(ex.getMessage(), ex);
                }

                return null;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

import javax.annotation.Resource;
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.sql.DataSource;
```

```
import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.entity.AuthorizationRoleEntity;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

/**
 * AuthorizationRole Repository implementation using java.sql.
    PreparedStatement
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 */
@ApplicationScoped
@Default
public class AuthorizationRoleRepository implements
   IAuthorizationRoleRepository {

        private static final Logger logger = Logger.getLogger(
           AccountRepository.class);

        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    DataSource dataSource;

        @Override
        public List<AuthorizationRoleEntity> getAll() {
                try (Connection connection = dataSource.getConnection()) {
                        final String query = "SELECT id, name FROM
                           authorization_role";

                        try (PreparedStatement stmt = connection.
                           prepareStatement(query)) {
                                List<AuthorizationRoleEntity> roles = new
                                   ArrayList<>();

                                ResultSet rs = stmt.executeQuery();

                                while(rs.next()) {
                                        AuthorizationRoleEntity authRole = new
                                             AuthorizationRoleEntity();
                                        authRole.setId(UUID.fromString(rs.
                                           getString("id")));
                                        authRole.setName(rs.getString("name"))
                                            ;

                                        roles.add(authRole);
                                }

                                return roles;
                        }
                } catch(Exception ex) {
                        logger.error(ex.getMessage(), ex);
```

```
        }

        return null;
}

@Override
public AuthorizationRoleEntity findById(UUID id) {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT id, name FROM
                    authorization_role WHERE id = ?";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {

                        stmt.setObject(1, id);
                        AuthorizationRoleEntity authRole = new
                            AuthorizationRoleEntity();

                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                authRole.setId(UUID.fromString(rs.
                                    getString("id")));
                                authRole.setName(rs.getString("name"))
                                    ;
                        }

                        return authRole;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public AuthorizationRoleEntity findByName(String name) {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT id, name FROM
                    authorization_role WHERE name = ?";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {

                        stmt.setObject(1, name);
                        AuthorizationRoleEntity authRole = new
                            AuthorizationRoleEntity();

                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                authRole.setId(UUID.fromString(rs.
                                    getString("id")));
```

```
                                        authRole.setName(rs.getString("name"))
                                            ;
                                }

                                return authRole;
                        }
                } catch(Exception ex) {
                        logger.error(ex.getMessage(), ex);
                }

                return null;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;

import javax.annotation.Resource;
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.sql.DataSource;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.EmployeeEntity;
import com.holidaysystem.enumeration.DepartmentEnum;
import com.holidaysystem.enumeration.EmployeeRoleEnum;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.model.EmployeeModel;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.time.LocalDateTime;
```

```
import java.util.ArrayList;
import java.util.List;

/**
 * Employee Repository implementation using java.sql.PreparedStatement
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class EmployeeRepository implements IEmployeeRepository {

        private static final Logger logger = Logger.getLogger(
            EmployeeRepository.class);

        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    DataSource dataSource;

        @Override
        public EmployeeEntity findById(UUID employeeId) {
                try (Connection connection = dataSource.getConnection()) {

                        final String query = "SELECT id, firstname, lastname,
                            role, department, years, accountid, created,
                            modified "
                                        + "FROM employee WHERE id = ?";

                        try (PreparedStatement stmt = connection.
                            prepareStatement(query)) {
                                stmt.setObject(1, employeeId);

                                EmployeeEntity employee = new EmployeeEntity()
                                    ;
                                ResultSet rs = stmt.executeQuery();
                                while(rs.next()) {
                                        employee.setId(UUID.fromString(rs.
                                            getString("id")));
                                        employee.setFirstName(rs.getString("
                                            firstname"));
                                        employee.setLastName(rs.getString("
                                            lastname"));
                                        employee.setRole(rs.getString("role"))
                                            ;
                                        employee.setDepartment(rs.getString("
                                            department"));
                                        employee.setYears(rs.getInt("years"));
                                        employee.setAccountId(UUID.fromString(
                                            rs.getString("accountid")));
                                        employee.setCreated(LocalDateTime.
                                            parse(rs.getString("created"),
                                            DateUtils.FORMATTER));
                                        employee.setModified(LocalDateTime.
                                            parse(rs.getString("modified"),
                                            DateUtils.FORMATTER));
```

```
                }

                return employee;
            }
        } catch(Exception ex) {
            logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public EmployeeEntity findByEmail(String email) {
        try (Connection connection = dataSource.getConnection()) {

            final String query = "SELECT id, firstname, lastname,
                role, department, accountid, years, "
                        + "created, modified "
                        + "FROM employee "
                        + "WHERE email = ?";

            try (PreparedStatement stmt = connection.
                prepareStatement(query)) {
                stmt.setString(1, email);

                EmployeeEntity employee = new EmployeeEntity()
                    ;
                ResultSet rs = stmt.executeQuery();
                while(rs.next()) {
                    employee.setId(UUID.fromString(rs.
                        getString("id")));
                    employee.setFirstName(rs.getString("
                        firstname"));
                    employee.setLastName(rs.getString("
                        lastname"));
                    employee.setRole(rs.getString("role"))
                        ;
                    employee.setDepartment(rs.getString("
                        department"));
                    employee.setAccountId(UUID.fromString(
                        rs.getString("accountid")));
                    employee.setYears(rs.getInt("years"));
                    employee.setCreated(LocalDateTime.
                        parse(rs.getString("created"),
                        DateUtils.FORMATTER));
                    employee.setModified(LocalDateTime.
                        parse(rs.getString("modified"),
                        DateUtils.FORMATTER));
                }

                return employee;
            }
        } catch(Exception ex) {
            logger.error(ex.getMessage(), ex);
```

```
        }

        return null;
}

@Override
public boolean save(EmployeeEntity employee) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "INSERT INTO employee (id,
                    firstname, lastname, role, department, accountid, "
                                + "years, created, modified) "
                                + "VALUES (?,?,?,?,?,?,?,?,?);";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, employee.getId());
                        stmt.setString(2, employee.getFirstName());
                        stmt.setString(3, employee.getLastName());
                        stmt.setObject(4, employee.getRole());
                        stmt.setObject(5, employee.getDepartment());
                        stmt.setObject(6, employee.getAccountId());
                        stmt.setObject(7, employee.getYears());
                        stmt.setObject(8, employee.getCreated());
                        stmt.setObject(9, employee.getModified());

                        if (stmt.executeUpdate() == 1) {
                                return true;
                        } else {
                                return false;
                        }
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return false;
}

@Override
public List<EmployeeEntity> getEmployees() {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT empl.id, empl.firstname,
                    empl.lastname,"
                                + " empl.role, empl.department, empl.
                                    accountid, "
                                + "empl.years, empl.created, empl.
                                    modified "
                                + "FROM employee empl;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        List<EmployeeEntity> employees = new ArrayList
                            <>();
```

```
                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                EmployeeEntity employee = new
                                    EmployeeEntity();
                                employee.setId(UUID.fromString(rs.
                                    getString("id")));
                                employee.setFirstName(rs.getString("
                                    firstname"));
                                employee.setLastName(rs.getString("
                                    lastname"));
                                employee.setRole(rs.getString("role"))
                                    ;
                                employee.setDepartment(rs.getString("
                                    department"));
                                employee.setAccountId(UUID.fromString(
                                    rs.getString("accountid")));
                                employee.setYears(rs.getInt("years"));
                                employee.setCreated(LocalDateTime.
                                    parse(rs.getString("created"),
                                    DateUtils.FORMATTER));
                                employee.setModified(LocalDateTime.
                                    parse(rs.getString("modified"),
                                    DateUtils.FORMATTER));

                                employees.add(employee);
                        }

                        return employees;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public List<EmployeeModel> getEmployeeModels() {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT empl.id, empl.firstname,
                    empl.lastname, " +
                                "empl.role, empl.department, empl.
                                    accountid, hd.totaldays, " +
                                "hd.takendays, hd.status, empl.years,
                                    acc.email " +
                                "FROM employee empl " +
                                "inner join account acc " +
                                "on empl.accountid = acc.id " +
                                "inner join holiday_details hd " +
                                "on hd.employeeid = empl.id;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
```

```
                        List<EmployeeModel> employees = new ArrayList
                            <>();
                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                EmployeeModel employee = new
                                    EmployeeModel();
                                employee.setId(UUID.fromString(rs.
                                    getString("id")));
                                employee.setFirstName(rs.getString("
                                    firstname"));
                                employee.setLastName(rs.getString("
                                    lastname"));
                                employee.setRole(EmployeeRoleEnum.
                                    valueOf(rs.getString("role")));
                                employee.setDepartment(DepartmentEnum.
                                    valueOf(rs.getString("department"))
                                    );
                                employee.setAccountId(UUID.fromString(
                                    rs.getString("accountid")));
                                employee.setEmail(rs.getString("email
                                    "));
                                employee.setYears(rs.getInt("years"));
                                employee.setTotalDays(rs.getInt("
                                    totaldays"));
                                employee.setTakenDays(rs.getInt("
                                    takendays"));
                                employee.setHolidayStatus(
                                    HolidayStatusEnum.valueOf(rs.
                                    getString("status")));

                                employees.add(employee);
                        }

                        return employees;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public List<EmployeeModel> getEmployeeModelsByDepartmentId(String
    department) {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT empl.id, empl.firstname,
                    empl.lastname, " +
                                "empl.role, empl.department, empl.
                                    accountid, hd.totaldays, " +
                                "hd.takendays, hd.status, empl.years,
                                    acc.email " +
                                "FROM employee empl " +
```

```
                                "inner join account acc " +
                                "on empl.accountid = acc.id " +
                                "inner join holiday_details hd " +
                                "on hd.employeeid = empl.id "
                                + "WHERE empl.department = ?;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setString(1, department);
                        List<EmployeeModel> employees = new ArrayList
                            <>();
                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                EmployeeModel employee = new
                                    EmployeeModel();
                                employee.setId(UUID.fromString(rs.
                                    getString("id")));
                                employee.setFirstName(rs.getString("
                                    firstname"));
                                employee.setLastName(rs.getString("
                                    lastname"));
                                employee.setRole(EmployeeRoleEnum.
                                    valueOf(rs.getString("role")));
                                employee.setDepartment(DepartmentEnum.
                                    valueOf(rs.getString("department"))
                                    );
                                employee.setAccountId(UUID.fromString(
                                    rs.getString("accountid")));
                                employee.setEmail(rs.getString("email
                                    "));
                                employee.setYears(rs.getInt("years"));
                                employee.setTotalDays(rs.getInt("
                                    totaldays"));
                                employee.setTakenDays(rs.getInt("
                                    takendays"));
                                employee.setHolidayStatus(
                                    HolidayStatusEnum.valueOf(rs.
                                    getString("status")));

                                employees.add(employee);
                        }

                        return employees;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public List<EmployeeModel> getEmployeeModelsOnHolidaysByDate(
```

```
LocalDateTime date) {
    try (Connection connection = dataSource.getConnection()) {
            final String query = "SELECT empl.id, empl.firstname,
                empl.lastname, " +
                        "empl.role, empl.department, empl.
                            accountid, hd.totaldays, " +
                        "hd.takendays, hd.status, empl.years,
                            acc.email " +
                        "FROM employee empl " +
                        "inner join account acc " +
                        "on empl.accountid = acc.id " +
                        "inner join holiday_details hd " +
                        "on hd.employeeid = empl.id " +
                        "inner join holiday_request hr " +
                        "on hr.employeeid = empl.id " +
                        "WHERE hr.startdate <= ? AND hr.
                            enddate >= ?;";

            try (PreparedStatement stmt = connection.
                prepareStatement(query)) {
                    stmt.setObject(1, date);
                    stmt.setObject(2, date);
                    List<EmployeeModel> employees = new ArrayList
                        <>();
                    ResultSet rs = stmt.executeQuery();

                    while(rs.next()) {
                            EmployeeModel employee = new
                                EmployeeModel();
                            employee.setId(UUID.fromString(rs.
                                getString("id")));
                            employee.setFirstName(rs.getString("
                                firstname"));
                            employee.setLastName(rs.getString("
                                lastname"));
                            employee.setRole(EmployeeRoleEnum.
                                valueOf(rs.getString("role")));
                            employee.setDepartment(DepartmentEnum.
                                valueOf(rs.getString("department"))
                                );
                            employee.setAccountId(UUID.fromString(
                                rs.getString("accountid")));
                            employee.setEmail(rs.getString("email
                                "));
                            employee.setYears(rs.getInt("years"));
                            employee.setTotalDays(rs.getInt("
                                totaldays"));
                            employee.setTakenDays(rs.getInt("
                                takendays"));
                            employee.setHolidayStatus(
                                HolidayStatusEnum.valueOf(rs.
                                getString("status")));

                            employees.add(employee);
```

```
                        }

                        return employees;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public List<EmployeeModel> getEmployeeModelsByDateAndHolidayStatus(
    LocalDateTime date, HolidayStatusEnum holidayStatus) {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT empl.id, empl.firstname,
                    empl.lastname, " +
                                "empl.role, empl.department, empl.
                                    accountid, hd.totaldays, " +
                                "hd.takendays, hd.status, empl.years,
                                    acc.email " +
                                "FROM employee empl " +
                                "inner join account acc " +
                                "on empl.accountid = acc.id " +
                                "inner join holiday_details hd " +
                                "on hd.employeeid = empl.id " +
                                "inner join holiday_request hr " +
                                "on hr.employeeid = empl.id " +
                                "WHERE hr.startdate <= ? AND hr.
                                    enddate >= ? AND "
                               + "hd.status = ?;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, date);
                        stmt.setObject(2, date);
                        stmt.setObject(3, holidayStatus.name());
                        List<EmployeeModel> employees = new ArrayList
                            <>();
                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                EmployeeModel employee = new
                                    EmployeeModel();
                                employee.setId(UUID.fromString(rs.
                                    getString("id")));
                                employee.setFirstName(rs.getString("
                                    firstname"));
                                employee.setLastName(rs.getString("
                                    lastname"));
                                employee.setRole(EmployeeRoleEnum.
                                    valueOf(rs.getString("role")));
                                employee.setDepartment(DepartmentEnum.
                                    valueOf(rs.getString("department"))
```

```
                                        );
                                    employee.setAccountId(UUID.fromString(
                                        rs.getString("accountid")));
                                    employee.setEmail(rs.getString("email
                                        "));
                                    employee.setYears(rs.getInt("years"));
                                    employee.setTotalDays(rs.getInt("
                                        totaldays"));
                                    employee.setTakenDays(rs.getInt("
                                        takendays"));
                                    employee.setHolidayStatus(
                                        HolidayStatusEnum.valueOf(rs.
                                        getString("status")));

                                    employees.add(employee);
                                }

                                return employees;
                            }
                        } catch(Exception ex) {
                                logger.error(ex.getMessage(), ex);
                        }

                        return null;
                }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;

import javax.annotation.Resource;
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.sql.DataSource;

import org.jboss.logging.Logger;
```

```
import com.holidaysystem.Constants;
import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.HolidayDetailsEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.model.HolidayRequestModel;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;

/**
 * HolidayDetails Repository implementation using java.sql.PreparedStatement
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class HolidayDetailsRepository implements IHolidayDetailsRepository {

        private static final Logger logger = Logger.getLogger(
           HolidayDetailsRepository.class);

        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    DataSource dataSource;

        @Override
        public HolidayDetailsEntity findById(UUID id) {
                try (Connection connection = dataSource.getConnection()) {

                        final String query = "SELECT id, employeeid, status,
                           totaldays, takendays, created, modified "
                                        + "FROM holiday_details "
                                        + "WHERE id = ?";

                        try (PreparedStatement stmt = connection.
                           prepareStatement(query)) {
                                stmt.setObject(1, id);

                                ResultSet rs = stmt.executeQuery();
                                while(rs.next()) {
                                        HolidayDetailsEntity entity = new
                                           HolidayDetailsEntity();
                                        entity.setId(UUID.fromString(rs.
                                           getString("id")));
                                        entity.setEmployeeId(UUID.fromString(
                                           rs.getString("employeeid")));
                                        entity.setTotalDays(rs.getInt("
                                           totaldays"));
                                        entity.setTakenDays(rs.getInt("
```

```
                                takendays"));
                        entity.setStatus(rs.getString("status
                            "));
                        entity.setCreated(LocalDateTime.parse(
                            rs.getString("created"), DateUtils.
                            FORMATTER));
                        entity.setModified(LocalDateTime.parse
                            (rs.getString("modified"),
                            DateUtils.FORMATTER));
                        return entity;
                    }

                    return null;
                }
            } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
            }

            return null;
    }

    @Override
    public HolidayDetailsEntity findByEmployeeId(UUID employeeId) {
            try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT id, employeeid, status,
                    totaldays, takendays, created, modified "
                                + "FROM holiday_details "
                                + "WHERE employeeId = ?";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, employeeId);

                        ResultSet rs = stmt.executeQuery();
                        while(rs.next()) {
                                HolidayDetailsEntity entity = new
                                    HolidayDetailsEntity();
                                entity.setId(UUID.fromString(rs.
                                    getString("id")));
                                entity.setEmployeeId(UUID.fromString(
                                    rs.getString("employeeid")));
                                entity.setTotalDays(rs.getInt("
                                    totaldays"));
                                entity.setTakenDays(rs.getInt("
                                    takendays"));
                                entity.setStatus(rs.getString("status
                                    "));
                                entity.setCreated(LocalDateTime.parse(
                                    rs.getString("created"), DateUtils.
                                    FORMATTER));
                                entity.setModified(LocalDateTime.parse
                                    (rs.getString("modified"),
                                    DateUtils.FORMATTER));
```

```
                        return entity;
                    }

                    return null;
                }
            } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
            }

            return null;
    }

    @Override
    public boolean save(HolidayDetailsEntity entity) {
            try (Connection connection = dataSource.getConnection()) {

                final String query = "INSERT INTO holiday_details (id,
                    employeeid, totaldays, takendays, "
                            + " status, created, modified) "
                            + "VALUES (?,?,?,?,?,?,?);";

                try (PreparedStatement stmt = connection.
                   prepareStatement(query)) {
                        stmt.setObject(1, entity.getId());
                        stmt.setObject(2, entity.getEmployeeId());
                        stmt.setObject(3, entity.getTotalDays());
                        stmt.setObject(4, entity.getTakenDays());
                        stmt.setObject(5, entity.getStatus());
                        stmt.setObject(6, entity.getCreated());
                        stmt.setObject(7, entity.getModified());

                        if (stmt.executeUpdate() == 1) {
                            return true;
                        } else {
                            return false;
                        }
                }
            } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
            }

            return false;
    }

    @Override
    public List<HolidayDetailsEntity> getHolidayDetails() {
            try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT id, employeeid, totaldays
                   , takendays, status, created, modified "
                            + " FROM holiday_details;";

                try (PreparedStatement stmt = connection.
```

```
                    prepareStatement(query)) {
                List<HolidayDetailsEntity> entities = new
                    ArrayList<>();
                ResultSet rs = stmt.executeQuery();
                while(rs.next()) {
                        HolidayDetailsEntity entity = new
                            HolidayDetailsEntity();

                        entity.setId(UUID.fromString(rs.
                            getString("id")));
                        entity.setEmployeeId(UUID.fromString(
                            rs.getString("employeeid")));
                        entity.setTotalDays(rs.getInt("
                            totaldays"));
                        entity.setTakenDays(rs.getInt("
                            takendays"));
                        entity.setStatus(rs.getString("status
                            "));
                        entity.setCreated(LocalDateTime.parse(
                            rs.getString("created"), DateUtils.
                            FORMATTER));
                        entity.setModified(LocalDateTime.parse
                            (rs.getString("modified"),
                            DateUtils.FORMATTER));

                        entities.add(entity);
                }

                return entities;
            }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public List<HolidayRequestModel> getApprovedAndByDate(LocalDateTime
    date, HolidayRequestStatusEnum requestStatus) {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT hr.id, hd.employeeid, hd.
                    takendays, hd.totaldays, " +
                                "hr.startdate, hr.enddate, hd.
                                    status as holidaystatus, "
                                + "hr.status as requeststatus
                                    " +
                        "FROM holiday_request hr " +
                                "inner join holiday_details hd
                                    " +
                                "on hd.employeeid = hr.
                                    employeeid " +
                        "WHERE hr.startdate <= ? AND hr.
                            enddate >= ? AND " +
```

```
                                        "hr.status = ?;";

                        try (PreparedStatement stmt = connection.
                            prepareStatement(query)) {
                                stmt.setObject(1, date);
                                stmt.setObject(2, date);
                                stmt.setObject(3, requestStatus.name());
                                List<HolidayRequestModel> requestModels = new
                                    ArrayList<>();
                                ResultSet rs = stmt.executeQuery();

                                while(rs.next()) {
                                        HolidayRequestModel model = new
                                            HolidayRequestModel();
                                        model.setId(UUID.fromString(rs.
                                            getString("id")));
                                        model.setEmployeeId(UUID.fromString(rs
                                            .getString("employeeid")));
                                        model.setTotalDays(rs.getInt("
                                            totaldays"));
                                        model.setTakenDays(rs.getInt("
                                            takendays"));
                                        model.setHolidayStatus(
                                            HolidayStatusEnum.valueOf(rs.
                                            getString("holidaystatus")));
                                        model.setRequestStatus(
                                            HolidayRequestStatusEnum.valueOf(rs
                                            .getString("requeststatus")));
                                        model.setStartDate(LocalDateTime.parse
                                            (rs.getString("startdate"),
                                            DateUtils.FORMATTER));
                                        model.setEndDate(LocalDateTime.parse(
                                            rs.getString("enddate"), DateUtils.
                                           FORMATTER));

                                        requestModels.add(model);
                                }

                                return requestModels;
                        }
                } catch(Exception ex) {
                        logger.error(ex.getMessage(), ex);
                }

                return null;
        }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
```

```
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;

import javax.annotation.Resource;
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.sql.DataSource;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.HolidayRequestEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.model.HolidayRequestModel;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;

/**
 * HolidayRequest Repository implementation using java.sql.PreparedStatement
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class HolidayRequestRepository implements IHolidayRequestRepository {

        private static final Logger logger = Logger.getLogger(
            HolidayRequestRepository.class);

        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    private DataSource dataSource;

        @Override
        public HolidayRequestEntity findById(UUID holidayRequestId) {
                try (Connection connection = dataSource.getConnection()) {
```

```
                final String query = "SELECT id, employeeid, status,
                    startdate, enddate, created, modified "
                                + "FROM holiday_request "
                                + "WHERE id = ?";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, holidayRequestId);

                        HolidayRequestEntity entity = new
                            HolidayRequestEntity();
                        ResultSet rs = stmt.executeQuery();
                        while(rs.next()) {
                                entity.setId(UUID.fromString(rs.
                                    getString("id")));
                                entity.setEmployeeId(UUID.fromString(
                                    rs.getString("employeeid")));
                                entity.setStatus(rs.getString("status
                                    "));
                                entity.setStartDate(LocalDateTime.
                                    parse(rs.getString("startdate"),
                                    DateUtils.FORMATTER));
                                entity.setEndDate(LocalDateTime.parse(
                                    rs.getString("enddate"), DateUtils.
                                    FORMATTER));
                                entity.setCreated(LocalDateTime.parse(
                                    rs.getString("created"), DateUtils.
                                    FORMATTER));
                                entity.setModified(LocalDateTime.parse
                                    (rs.getString("modified"),
                                    DateUtils.FORMATTER));
                        }

                        return entity;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}


@Override
public boolean save(HolidayRequestEntity holidayRequestEntity) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "INSERT INTO holiday_request (id,
                     employeeid, status, startdate, enddate, created,
                    modified) "
                                + "VALUES (?,?,?,?,?,?,?);";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
```

```
                        stmt.setObject(1, holidayRequestEntity.getId()
                            );
                        stmt.setObject(2, holidayRequestEntity.
                            getEmployeeId());
                        stmt.setString(3, holidayRequestEntity.
                            getStatus());
                        stmt.setObject(4, holidayRequestEntity.
                            getStartDate());
                        stmt.setObject(5, holidayRequestEntity.
                            getEndDate());
                        stmt.setObject(6, holidayRequestEntity.
                            getCreated());
                        stmt.setObject(7, holidayRequestEntity.
                            getModified());

                        if (stmt.executeUpdate() == 1) {
                            return true;
                        } else {
                            return false;
                        }
                }

        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return false;
}


@Override
public List<HolidayRequestEntity> getHolidayRequests() {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT id, employeeid, status,
                    startdate, enddate, created, modified "
                                + "FROM holiday_request";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        List<HolidayRequestEntity> holidayRequests =
                            new ArrayList<>();
                        ResultSet rs = stmt.executeQuery();
                        while(rs.next()) {
                                HolidayRequestEntity entity = new
                                    HolidayRequestEntity();
                                entity.setId(UUID.fromString(rs.
                                    getString("id")));
                                entity.setEmployeeId(UUID.fromString(
                                    rs.getString("employeeid")));
                                entity.setStatus(rs.getString("status
                                    "));
                                entity.setStartDate(LocalDateTime.
                                    parse(rs.getString("startdate"),
```

```
                                    DateUtils.FORMATTER));
                            entity.setEndDate(LocalDateTime.parse(
                                rs.getString("enddate"), DateUtils.
                                FORMATTER));
                            entity.setCreated(LocalDateTime.parse(
                                rs.getString("created"), DateUtils.
                                FORMATTER));
                            entity.setModified(LocalDateTime.parse
                                (rs.getString("modified"),
                                DateUtils.FORMATTER));

                            holidayRequests.add(entity);
                        }

                        return holidayRequests;
                    }
            } catch(Exception ex) {
                    logger.error(ex.getMessage(), ex);
            }

            return null;
    }


    @Override
    public HolidayRequestEntity update(UUID id, HolidayRequestEntity
        holidayRequestEntity) {
            try (Connection connection = dataSource.getConnection()) {

                    final String query = "UPDATE holiday_request SET
                        status = ?, startdate = ?, enddate = ?, modified =
                        ?, employeeid = ? "
                                    + "WHERE id= ?;";

                    try (PreparedStatement stmt = connection.
                        prepareStatement(query)) {
                            stmt.setString(1, holidayRequestEntity.
                                getStatus());
                            stmt.setObject(2, holidayRequestEntity.
                                getStartDate());
                            stmt.setObject(3, holidayRequestEntity.
                                getEndDate());
                            stmt.setObject(4, holidayRequestEntity.
                                getModified());
                            stmt.setObject(5, holidayRequestEntity.
                                getEmployeeId());
                            stmt.setObject(6, id);

                            if (stmt.executeUpdate() == 1) {
                                return holidayRequestEntity;
                            } else {
                                return null;
                            }
                    }
```

```
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}


@Override
public List<HolidayRequestEntity> getHolidayRequestsByStatus(
    HolidayRequestStatusEnum requestStatus) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT id, employeeid, status,
                    startdate, enddate, created, modified "
                                + "FROM holiday_request "
                                + "WHERE status = ?";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, requestStatus.name());

                        List<HolidayRequestEntity> holidayRequests =
                            new ArrayList<>();
                        ResultSet rs = stmt.executeQuery();
                        while(rs.next()) {
                                HolidayRequestEntity entity = new
                                    HolidayRequestEntity();
                                entity.setId(UUID.fromString(rs.
                                    getString("id")));
                                entity.setEmployeeId(UUID.fromString(
                                    rs.getString("employeeid")));
                                entity.setStatus(rs.getString("status
                                    "));
                                entity.setStartDate(LocalDateTime.
                                    parse(rs.getString("startdate"),
                                    DateUtils.FORMATTER));
                                entity.setEndDate(LocalDateTime.parse(
                                    rs.getString("enddate"), DateUtils.
                                    FORMATTER));
                                entity.setCreated(LocalDateTime.parse(
                                    rs.getString("created"), DateUtils.
                                    FORMATTER));
                                entity.setModified(LocalDateTime.parse
                                    (rs.getString("modified"),
                                    DateUtils.FORMATTER));

                                holidayRequests.add(entity);
                        }

                        return holidayRequests;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
```

```
        }

        return null;
}

@Override
public List<HolidayRequestModel> getHolidayRequestModels() {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT hr.id, hr.employeeid, hr.
                    status, " +
                                "hr.startdate, hr.enddate, " +
                                "hd.totaldays, hd.takendays, empl.
                                    years " +
                                "FROM holiday_request hr " +
                                "inner join employee empl " +
                                "on empl.id = hr.employeeid " +
                                "inner join holiday_details hd " +
                                "on hd.employeeid = empl.id "
                                + "ORDER BY hr.created "
                                + "limit 100;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {

                        List<HolidayRequestModel> models = new
                            ArrayList<>();
                        ResultSet rs = stmt.executeQuery();


                        while(rs.next()) {

                                HolidayRequestModel model = new
                                    HolidayRequestModel();
                                model.setId(UUID.fromString(rs.
                                    getString("id")));
                                model.setEmployeeId(UUID.fromString(rs
                                    .getString("employeeid")));
                                model.setRequestStatus(
                                    HolidayRequestStatusEnum.valueOf(rs
                                    .getString("status")));
                                model.setStartDate(LocalDateTime.parse
                                    (rs.getString("startdate"),
                                    DateUtils.FORMATTER));
                                model.setEndDate(LocalDateTime.parse(
                                    rs.getString("enddate"), DateUtils.
                                    FORMATTER));
                                model.setYears(rs.getInt("years"));
                                model.setTotalDays(rs.getInt("
                                    totaldays"));
                                model.setTakenDays(rs.getInt("
                                    takendays"));

                                models.add(model);
                        }
```

```
                        return models;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public List<HolidayRequestModel> getHolidayRequestModels(int offset,
    int limit) {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT hr.id, hr.employeeid, hr.
                    status, " +
                                "hr.startdate, hr.enddate, " +
                                "hd.totaldays, hd.takendays, empl.
                                    years " +
                                "FROM holiday_request hr " +
                                "inner join employee empl " +
                                "on empl.id = hr.employeeid " +
                                "inner join holiday_details hd " +
                                "on hd.employeeid = empl.id "
                                + "ORDER BY hr.created "
                                + "limit ? offset ?;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setInt(1, limit);
                        stmt.setInt(2, offset);

                        List<HolidayRequestModel> models = new
                            ArrayList<>();
                        ResultSet rs = stmt.executeQuery();


                        while(rs.next()) {

                                HolidayRequestModel model = new
                                    HolidayRequestModel();
                                model.setId(UUID.fromString(rs.
                                    getString("id")));
                                model.setEmployeeId(UUID.fromString(rs
                                    .getString("employeeid")));
                                model.setRequestStatus(
                                    HolidayRequestStatusEnum.valueOf(rs
                                    .getString("status")));
                                model.setStartDate(LocalDateTime.parse
                                    (rs.getString("startdate"),
                                    DateUtils.FORMATTER));
                                model.setEndDate(LocalDateTime.parse(
                                    rs.getString("enddate"), DateUtils.
                                    FORMATTER));
```

```
                                model.setYears(rs.getInt("years"));
                                model.setTotalDays(rs.getInt("
                                    totaldays"));
                                model.setTakenDays(rs.getInt("
                                    takendays"));

                                models.add(model);
                        }

                        return models;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public List<HolidayRequestModel> getHolidayRequestModelsByStatus(
    HolidayRequestStatusEnum requestStatus) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT hr.id, hr.employeeid, hr.
                    status, " +
                                "hr.startdate, hr.enddate, " +
                                "hd.totaldays, hd.takendays, empl.
                                    years " +
                                "FROM holiday_request hr " +
                                "inner join employee empl " +
                                "on empl.id = hr.employeeid " +
                                "inner join holiday_details hd " +
                                "on hd.employeeid = empl.id " +
                                "WHERE hr.status = ?;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, requestStatus.name());

                        List<HolidayRequestModel> models = new
                            ArrayList<>();
                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {

                                HolidayRequestModel model = new
                                    HolidayRequestModel();
                                model.setId(UUID.fromString(rs.
                                    getString("id")));
                                model.setEmployeeId(UUID.fromString(rs
                                    .getString("employeeid")));
                                model.setRequestStatus(
                                    HolidayRequestStatusEnum.valueOf(rs
                                    .getString("status")));
```

```
                                        model.setStartDate(LocalDateTime.parse
                                            (rs.getString("startdate"),
                                            DateUtils.FORMATTER));
                                        model.setEndDate(LocalDateTime.parse(
                                            rs.getString("enddate"), DateUtils.
                                            FORMATTER));
                                        model.setYears(rs.getInt("years"));
                                        model.setTotalDays(rs.getInt("
                                            totaldays"));
                                        model.setTakenDays(rs.getInt("
                                            takendays"));

                                        models.add(model);
                                }

                                return models;
                        }
                } catch(Exception ex) {
                        logger.error(ex.getMessage(), ex);
                }

                return null;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;

import com.holidaysystem.entity.AccountEntity;

/**
 * Interface for Account repository, which can be supported by
 * all implementations of the interface
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
```

```
public interface IAccountRepository {
        /**
         * Finds account by account id
         * @param accountId UUID account id
         * @return account entity
         */
        AccountEntity findById(UUID accountId);
    /**
     * Finds account by email
     * @param email account email
     * @return account entity
     */
        AccountEntity findByEmail(String email);
        /**
         * Finds account by email and password
         * @param email account email
         * @param password account password
         * @return account entity
         */
        AccountEntity findByEmailAndPassword(String email, String password);
    /**
     * Adds a new account into persistence storage
     * @param account new account entity
     * @return saved account entity
     */
    boolean save(AccountEntity account);
    /**
     * Generated hashed password of the password
     * @param password account password
     * @return hashed password
     */
    String generateHashedPassword(String password);
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.List;
```

```
import java.util.UUID;

import com.holidaysystem.entity.AuthorizationRoleEntity;

/**
 * Interface for AuthorizationRole repository, which can be supported by
 * all implementations of the interface
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IAuthorizationRoleRepository {
        /**
         * Gets all authorization roles
         * @return list of AuthorizationRoleEntity
         */
        List<AuthorizationRoleEntity> getAll();
        /**
         * Gets authorization role by id
         * @param id authorization role id
         * @return AuthorizationRoleEntity
         */
        AuthorizationRoleEntity findById(UUID id);
        /**
         * Gets authorization role by name
         * @param name authorization role name
         * @return AuthorizationRoleEntity
         */
        AuthorizationRoleEntity findByName(String name);
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;
import java.time.LocalDateTime;
import java.util.List;
import com.holidaysystem.entity.EmployeeEntity;
import com.holidaysystem.enumeration.HolidayStatusEnum;
```

```
import com.holidaysystem.model.EmployeeModel;

/**
 * Interface for Employee repository, which can be supported by
 * all implementations of the interface
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IEmployeeRepository {
        /**
         * Finds employee entity by id
         * @param employeeId UUID employee id
         * @return employee entity
         */
        EmployeeEntity findById(UUID employeeId);
    /**
     * Finds employee entity by email
     * @param email
     * @return employee entity
     */
    EmployeeEntity findByEmail(String email);
    /**
     * Adds a new employee
     * @param employeeEntity employee entity
     * @return true is added, false is failed
     */
    boolean save(EmployeeEntity employeeEntity);
    /**
     * Gets all employee entities
     * @return list of employee entities
     */
    List<EmployeeEntity> getEmployees();
    /**
     * Gets all employee models
     * @return list of employee models
     */
    List<EmployeeModel> getEmployeeModels();
    /**
     * Gets employees from specific department
     * @param department department
     * @return list of employee models
     */
    List<EmployeeModel> getEmployeeModelsByDepartmentId(String department);
    /**
     * Gets employees on holidays on specific date
     * @param date the date
     * @return list of employee models on holidays
     */
    List<EmployeeModel> getEmployeeModelsOnHolidaysByDate(LocalDateTime date);
    /**
     * Gets employees on holidays or on duty on specific date
     * @param date the date
     * @param holidayStatus the status of HolidayStatusEnum
     * @return list of employee models
```

```
     */
    List<EmployeeModel> getEmployeeModelsByDateAndHolidayStatus(LocalDateTime
        date, HolidayStatusEnum holidayStatus);
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;
import java.time.LocalDateTime;
import java.util.List;
import com.holidaysystem.entity.HolidayDetailsEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.model.HolidayRequestModel;

/**
 * Interface for Holiday Details repository, which can be supported by
 * all implementations of the interface
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IHolidayDetailsRepository {
        /**
         * Finds holiday details entity by id
         * @param Id UUID of holiday details entity
         * @return HolidayDetailsEntity entity
         */
        HolidayDetailsEntity findById(UUID Id);
        /**
         * Finds employee by employee Id
         * @param employeeId
         * @return HolidayDetailsEntity entity
         */
        HolidayDetailsEntity findByEmployeeId(UUID employeeId);
    /**
     * Adds a new holiday detail entity
     * @param holidayDetails
     * @return true is saved, false is failed
```

```
     */
    boolean save(HolidayDetailsEntity holidayDetails);
    /**
     * Fetches all holiday detail entities
     * @return list of HolidayDetailsEntity entities
     */
    List<HolidayDetailsEntity> getHolidayDetails();
    /**
     * Fetches HolidayRequest models
     * @param date LocalDateTime
     * @param requestStatus HolidayRequestStatusEnum enumeration
     * @return list of HolidayRequestModel models
     */
    List<HolidayRequestModel> getApprovedAndByDate(LocalDateTime date,
        HolidayRequestStatusEnum requestStatus);
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;
import java.util.List;
import com.holidaysystem.entity.HolidayRequestEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.model.HolidayRequestModel;

/**
 * Interface for Holiday Request repository, which can be supported by
 * all implementations of the interface
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IHolidayRequestRepository {
        List<HolidayRequestEntity> getHolidayRequests();
        List<HolidayRequestModel> getHolidayRequestModels();
        List<HolidayRequestModel> getHolidayRequestModels(int offset, int
            limit);
        List<HolidayRequestEntity> getHolidayRequestsByStatus(
```

```
            HolidayRequestStatusEnum requestStatus);
        List<HolidayRequestModel> getHolidayRequestModelsByStatus(
            HolidayRequestStatusEnum requestStatus);
        HolidayRequestEntity findById(UUID holidayRequestId);
        boolean save(HolidayRequestEntity holidayRequestEntity);
        HolidayRequestEntity update(UUID id, HolidayRequestEntity
            holidayRequestEntity);
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;

import javax.annotation.Resource;
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.sql.DataSource;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.RequestAlertEntity;

import jakarta.inject.Named;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;

/**
 * RequestAlert Repository implementation using java.sql.PreparedStatement
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
```

```
 */
@ApplicationScoped
@Default
public class RequestAlertRepository implements IRequestAlertRepository {

        private static final Logger logger = Logger.getLogger(
            RequestAlertRepository.class);

        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    private DataSource dataSource;

        @Override
        public RequestAlertEntity findById(UUID requestAlertId) {
                try (Connection connection = dataSource.getConnection()) {

                        final String query = "SELECT id, requestid, date,
                            created, modified FROM request_alert_queue WHERE id
                             = ?";

                        try (PreparedStatement stmt = connection.
                            prepareStatement(query)) {
                                stmt.setObject(1, requestAlertId);

                                RequestAlertEntity entity = new
                                    RequestAlertEntity();
                                ResultSet rs = stmt.executeQuery();
                                while(rs.next()) {
                                        entity.setId(UUID.fromString(rs.
                                            getString("id")));
                                        entity.setRequestId(UUID.fromString(rs
                                            .getString("requestid")));
                                        entity.setDate(LocalDateTime.parse(rs.
                                            getString("startdate"), DateUtils.
                                            FORMATTER));
                                        entity.setCreated(LocalDateTime.parse(
                                            rs.getString("created"), DateUtils.
                                            FORMATTER));
                                        entity.setModified(LocalDateTime.parse
                                            (rs.getString("modified"),
                                            DateUtils.FORMATTER));
                                }

                                return entity;
                        }
                } catch(Exception ex) {
                        logger.error(ex.getMessage(), ex);
                }

                return null;
        }


        @Override
        public boolean save(RequestAlertEntity requestAlertEntity) {
```

```
        try (Connection connection = dataSource.getConnection()) {

                final String query = "INSERT INTO request_alert_queue
                    (id, requestid, date, created, modified) "
                            + "VALUES (?,?,?,?,?);";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, requestAlertEntity.getId());
                        stmt.setObject(2, requestAlertEntity.
                            getRequestId());
                        stmt.setObject(3, requestAlertEntity.getDate()
                            );
                        stmt.setObject(4, requestAlertEntity.
                            getCreated());
                        stmt.setObject(5, requestAlertEntity.
                            getModified());

                        if (stmt.executeUpdate() == 1) {
                            return true;
                        } else {
                            return false;
                        }
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return false;
}


@Override
public List<RequestAlertEntity> getRequestAlerts() {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT id, requestid, date,
                    created, modified FROM request_alert_queue";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        List<RequestAlertEntity> requestAlerts = new
                            ArrayList<>();
                        ResultSet rs = stmt.executeQuery();
                        while(rs.next()) {
                                RequestAlertEntity entity = new
                                    RequestAlertEntity();
                                entity.setId(UUID.fromString(rs.
                                    getString("id")));
                                entity.setRequestId(UUID.fromString(rs
                                    .getString("requestid")));
                                entity.setDate(LocalDateTime.parse(rs.
                                    getString("date"), DateUtils.
                                    FORMATTER));
```

```
                                        entity.setCreated(LocalDateTime.parse(
                                            rs.getString("created"), DateUtils.
                                            FORMATTER));
                                        entity.setModified(LocalDateTime.parse
                                            (rs.getString("modified"),
                                            DateUtils.FORMATTER));

                                        requestAlerts.add(entity);
                                }

                                return requestAlerts;
                        }
                } catch(Exception ex) {
                        logger.error(ex.getMessage(), ex);
                }

                return null;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;
import java.util.List;
import com.holidaysystem.entity.RequestAlertEntity;

/**
 * Interface for Request Alert repository, which can be supported by
 * all implementations of the interface
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IRequestAlertRepository {
        List<RequestAlertEntity> getRequestAlerts();
        RequestAlertEntity findById(UUID holidayRequestId);
        boolean save(RequestAlertEntity holidayRequestEntity);
}
```

] package com.holidaysystem.repository;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *           http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;

import javax.annotation.Resource;
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.sql.DataSource;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.AccountEntity;

import jakarta.inject.Named;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.time.LocalDateTime;

/**
 * Account Repository implementation using java.sql.PreparedStatement
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 */
@ApplicationScoped
@Default
public class AccountRepository implements IAccountRepository {

        private static final Logger logger = Logger.getLogger(
            AccountRepository.class);

        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    DataSource dataSource;
```

```
@Override
public AccountEntity findById(UUID accountId) {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT acc.id, acc.email, acc.
                    password, acc.active, "
                                + "acc.auth_roleid, acc.created, acc.
                                    modified "
                                + "FROM account acc "
                                + "WHERE acc.id = ?";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, accountId);

                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                AccountEntity account = new
                                    AccountEntity();
                                account.setId(UUID.fromString(rs.
                                    getString("id")));
                                account.setPassword(rs.getString("
                                    password"));
                                account.setEmail(rs.getString("email")
                                    );
                                account.setActive(rs.getBoolean("
                                    active"));
                                account.setAuthRoleId(UUID.fromString(
                                    rs.getString("auth_roleid")));
                                account.setCreated(LocalDateTime.parse
                                    (rs.getString("created"), DateUtils
                                    .FORMATTER));
                                account.setModified(LocalDateTime.
                                    parse(rs.getString("modified"),
                                    DateUtils.FORMATTER));

                                return account;
                        }

                        return null;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public AccountEntity findByEmail(String email) {

        try (Connection connection = dataSource.getConnection()) {

                String query = "SELECT acc.id, acc.email, acc.password
```

```
                    , acc.active, acc.auth_roleid, acc.created, acc.
                    modified "
                            + "FROM account acc "
                            + "WHERE acc.email = ?";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                    stmt.setString(1, email);

                    ResultSet rs = stmt.executeQuery();
                    while(rs.next()) {
                            AccountEntity account = new
                                AccountEntity();
                            account.setId(UUID.fromString(rs.
                                getString("id")));
                            account.setPassword(rs.getString("
                                password"));
                            account.setEmail(rs.getString("email")
                                );
                            account.setActive(rs.getBoolean("
                                active"));
                            account.setAuthRoleId(UUID.fromString(
                                rs.getString("auth_roleid")));
                            account.setCreated(LocalDateTime.parse
                                (rs.getString("created"), DateUtils
                                .FORMATTER));
                            account.setModified(LocalDateTime.
                                parse(rs.getString("modified"),
                                DateUtils.FORMATTER));

                            return account;
                    }

                    return null;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public AccountEntity findByEmailAndPassword(String email, String
    password) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT acc.id, acc.email, acc.
                    password, acc.active, acc.auth_roleid, acc.created,
                     acc.modified "
                            + "FROM account acc "
                            + "WHERE acc.email = ? AND "
                            + "crypt(?, acc.password) = acc.
                                password";
```

```
                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setString(1, email);
                        stmt.setString(2, password);

                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                AccountEntity account = new
                                    AccountEntity();
                                account.setId(UUID.fromString(rs.
                                    getString("id")));
                                account.setPassword(rs.getString("
                                    password"));
                                account.setEmail(rs.getString("email")
                                    );
                                account.setActive(rs.getBoolean("
                                    active"));
                                account.setAuthRoleId(UUID.fromString(
                                    rs.getString("auth_roleid")));
                                account.setCreated(LocalDateTime.parse
                                    (rs.getString("created"), DateUtils
                                    .FORMATTER));
                                account.setModified(LocalDateTime.
                                    parse(rs.getString("modified"),
                                    DateUtils.FORMATTER));

                                return account;
                        }
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public boolean save(AccountEntity account) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "INSERT INTO account (id, email,
                    password, active, auth_roleid, created, modified) "
                                + "VALUES (?,?,?,?,?,?,?);";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, account.getId());
                        stmt.setString(2, account.getEmail());
                        stmt.setString(3, account.getPassword());
                        stmt.setObject(4, account.getActive());
                        stmt.setObject(5, account.getAuthRoleId());
                        stmt.setObject(6, account.getCreated());
```

```
                        stmt.setObject(7, account.getModified());

                        if (stmt.executeUpdate() == 1) {
                            return true;
                        } else {
                            return false;
                        }
                    }
                } catch(Exception ex) {
                    logger.error(ex.getMessage(), ex);
                }

                return false;
        }

    public String generateHashedPassword(String password) {
                try (Connection connection = dataSource.getConnection()) {

                    final String query = "SELECT crypt(?, gen_salt('bf',
                        8))";

                    try (PreparedStatement stmt = connection.
                        prepareStatement(query)) {
                            stmt.setObject(1, password);

                            ResultSet rs = stmt.executeQuery();

                            String hashedPass = "";

                            while(rs.next()) {
                                    hashedPass = rs.getString(1);
                            }

                            return hashedPass;
                    }
                } catch(Exception ex) {
                    logger.error(ex.getMessage(), ex);
                }

                return null;
        }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
```

```
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

import javax.annotation.Resource;
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.sql.DataSource;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.entity.AuthorizationRoleEntity;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

/**
 * AuthorizationRole Repository implementation using java.sql.
    PreparedStatement
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 */
@ApplicationScoped
@Default
public class AuthorizationRoleRepository implements
   IAuthorizationRoleRepository {

        private static final Logger logger = Logger.getLogger(
           AccountRepository.class);

        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    DataSource dataSource;

        @Override
        public List<AuthorizationRoleEntity> getAll() {
                try (Connection connection = dataSource.getConnection()) {
                        final String query = "SELECT id, name FROM
                           authorization_role";

                        try (PreparedStatement stmt = connection.
                           prepareStatement(query)) {
                                List<AuthorizationRoleEntity> roles = new
                                   ArrayList<>();

                                ResultSet rs = stmt.executeQuery();
```

```
                        while(rs.next()) {
                                AuthorizationRoleEntity authRole = new
                                    AuthorizationRoleEntity();
                                authRole.setId(UUID.fromString(rs.
                                    getString("id")));
                                authRole.setName(rs.getString("name"))
                                    ;

                                roles.add(authRole);
                        }

                        return roles;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public AuthorizationRoleEntity findById(UUID id) {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT id, name FROM
                    authorization_role WHERE id = ?";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {

                        stmt.setObject(1, id);
                        AuthorizationRoleEntity authRole = new
                            AuthorizationRoleEntity();

                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                authRole.setId(UUID.fromString(rs.
                                    getString("id")));
                                authRole.setName(rs.getString("name"))
                                    ;
                        }

                        return authRole;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public AuthorizationRoleEntity findByName(String name) {
```

```
                try (Connection connection = dataSource.getConnection()) {
                        final String query = "SELECT id, name FROM
                            authorization_role WHERE name = ?";

                        try (PreparedStatement stmt = connection.
                            prepareStatement(query)) {

                                stmt.setObject(1, name);
                                AuthorizationRoleEntity authRole = new
                                    AuthorizationRoleEntity();

                                ResultSet rs = stmt.executeQuery();

                                while(rs.next()) {
                                        authRole.setId(UUID.fromString(rs.
                                            getString("id")));
                                        authRole.setName(rs.getString("name"))
                                            ;
                                }

                                return authRole;
                        }
                } catch(Exception ex) {
                        logger.error(ex.getMessage(), ex);
                }

                return null;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;

import javax.annotation.Resource;
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
```

```
import javax.sql.DataSource;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.EmployeeEntity;
import com.holidaysystem.enumeration.DepartmentEnum;
import com.holidaysystem.enumeration.EmployeeRoleEnum;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.model.EmployeeModel;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;

/**
 * Employee Repository implementation using java.sql.PreparedStatement
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class EmployeeRepository implements IEmployeeRepository {

        private static final Logger logger = Logger.getLogger(
           EmployeeRepository.class);

        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    DataSource dataSource;

        @Override
        public EmployeeEntity findById(UUID employeeId) {
                try (Connection connection = dataSource.getConnection()) {

                        final String query = "SELECT id, firstname, lastname,
                           role, department, years, accountid, created,
                           modified "
                                        + "FROM employee WHERE id = ?";

                        try (PreparedStatement stmt = connection.
                           prepareStatement(query)) {
                                stmt.setObject(1, employeeId);

                                EmployeeEntity employee = new EmployeeEntity()
                                   ;
                                ResultSet rs = stmt.executeQuery();
                                while(rs.next()) {
                                        employee.setId(UUID.fromString(rs.
                                           getString("id")));
                                        employee.setFirstName(rs.getString("
```

```
                                        firstname"));
                                employee.setLastName(rs.getString("
                                        lastname"));
                                employee.setRole(rs.getString("role"))
                                        ;
                                employee.setDepartment(rs.getString("
                                        department"));
                                employee.setYears(rs.getInt("years"));
                                employee.setAccountId(UUID.fromString(
                                        rs.getString("accountid")));
                                employee.setCreated(LocalDateTime.
                                        parse(rs.getString("created"),
                                        DateUtils.FORMATTER));
                                employee.setModified(LocalDateTime.
                                        parse(rs.getString("modified"),
                                        DateUtils.FORMATTER));
                        }

                        return employee;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public EmployeeEntity findByEmail(String email) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT id, firstname, lastname,
                        role, department, accountid, years, "
                                + "created, modified "
                                + "FROM employee "
                                + "WHERE email = ?";

                try (PreparedStatement stmt = connection.
                        prepareStatement(query)) {
                        stmt.setString(1, email);

                        EmployeeEntity employee = new EmployeeEntity()
                                ;
                        ResultSet rs = stmt.executeQuery();
                        while(rs.next()) {
                                employee.setId(UUID.fromString(rs.
                                        getString("id")));
                                employee.setFirstName(rs.getString("
                                        firstname"));
                                employee.setLastName(rs.getString("
                                        lastname"));
                                employee.setRole(rs.getString("role"))
                                        ;
                                employee.setDepartment(rs.getString("
```

```
                                    department"));
                                employee.setAccountId(UUID.fromString(
                                    rs.getString("accountid")));
                                employee.setYears(rs.getInt("years"));
                                employee.setCreated(LocalDateTime.
                                    parse(rs.getString("created"),
                                    DateUtils.FORMATTER));
                                employee.setModified(LocalDateTime.
                                    parse(rs.getString("modified"),
                                    DateUtils.FORMATTER));
                        }

                        return employee;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public boolean save(EmployeeEntity employee) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "INSERT INTO employee (id,
                    firstname, lastname, role, department, accountid, "
                                + "years, created, modified) "
                                + "VALUES (?,?,?,?,?,?,?,?,?);";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, employee.getId());
                        stmt.setString(2, employee.getFirstName());
                        stmt.setString(3, employee.getLastName());
                        stmt.setObject(4, employee.getRole());
                        stmt.setObject(5, employee.getDepartment());
                        stmt.setObject(6, employee.getAccountId());
                        stmt.setObject(7, employee.getYears());
                        stmt.setObject(8, employee.getCreated());
                        stmt.setObject(9, employee.getModified());

                        if (stmt.executeUpdate() == 1) {
                                return true;
                        } else {
                                return false;
                        }
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return false;
}
```

```
@Override
public List<EmployeeEntity> getEmployees() {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT empl.id, empl.firstname,
                    empl.lastname,"
                                + " empl.role, empl.department, empl.
                                    accountid, "
                                + "empl.years, empl.created, empl.
                                    modified "
                                + "FROM employee empl;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        List<EmployeeEntity> employees = new ArrayList
                            <>();
                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                EmployeeEntity employee = new
                                    EmployeeEntity();
                                employee.setId(UUID.fromString(rs.
                                    getString("id")));
                                employee.setFirstName(rs.getString("
                                    firstname"));
                                employee.setLastName(rs.getString("
                                    lastname"));
                                employee.setRole(rs.getString("role"))
                                    ;
                                employee.setDepartment(rs.getString("
                                    department"));
                                employee.setAccountId(UUID.fromString(
                                    rs.getString("accountid")));
                                employee.setYears(rs.getInt("years"));
                                employee.setCreated(LocalDateTime.
                                    parse(rs.getString("created"),
                                    DateUtils.FORMATTER));
                                employee.setModified(LocalDateTime.
                                    parse(rs.getString("modified"),
                                    DateUtils.FORMATTER));

                                employees.add(employee);
                        }

                        return employees;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
```

```
public List<EmployeeModel> getEmployeeModels() {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT empl.id, empl.firstname,
                    empl.lastname, " +
                                "empl.role, empl.department, empl.
                                    accountid, hd.totaldays, " +
                                "hd.takendays, hd.status, empl.years,
                                    acc.email " +
                                "FROM employee empl " +
                                "inner join account acc " +
                                "on empl.accountid = acc.id " +
                                "inner join holiday_details hd " +
                                "on hd.employeeid = empl.id;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        List<EmployeeModel> employees = new ArrayList
                            <>();
                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                EmployeeModel employee = new
                                    EmployeeModel();
                                employee.setId(UUID.fromString(rs.
                                    getString("id")));
                                employee.setFirstName(rs.getString("
                                    firstname"));
                                employee.setLastName(rs.getString("
                                    lastname"));
                                employee.setRole(EmployeeRoleEnum.
                                    valueOf(rs.getString("role")));
                                employee.setDepartment(DepartmentEnum.
                                    valueOf(rs.getString("department"))
                                    );
                                employee.setAccountId(UUID.fromString(
                                    rs.getString("accountid")));
                                employee.setEmail(rs.getString("email
                                    "));
                                employee.setYears(rs.getInt("years"));
                                employee.setTotalDays(rs.getInt("
                                    totaldays"));
                                employee.setTakenDays(rs.getInt("
                                    takendays"));
                                employee.setHolidayStatus(
                                    HolidayStatusEnum.valueOf(rs.
                                    getString("status")));

                                employees.add(employee);
                        }

                        return employees;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
```

```
        }

        return null;
    }

    @Override
    public List<EmployeeModel> getEmployeeModelsByDepartmentId(String
        department) {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT empl.id, empl.firstname,
                    empl.lastname, " +
                                "empl.role, empl.department, empl.
                                    accountid, hd.totaldays, " +
                                "hd.takendays, hd.status, empl.years,
                                    acc.email " +
                                "FROM employee empl " +
                                "inner join account acc " +
                                "on empl.accountid = acc.id " +
                                "inner join holiday_details hd " +
                                "on hd.employeeid = empl.id "
                                + "WHERE empl.department = ?;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setString(1, department);
                        List<EmployeeModel> employees = new ArrayList
                            <>();
                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                EmployeeModel employee = new
                                    EmployeeModel();
                                employee.setId(UUID.fromString(rs.
                                    getString("id")));
                                employee.setFirstName(rs.getString("
                                    firstname"));
                                employee.setLastName(rs.getString("
                                    lastname"));
                                employee.setRole(EmployeeRoleEnum.
                                    valueOf(rs.getString("role")));
                                employee.setDepartment(DepartmentEnum.
                                    valueOf(rs.getString("department"))
                                    );
                                employee.setAccountId(UUID.fromString(
                                    rs.getString("accountid")));
                                employee.setEmail(rs.getString("email
                                    "));
                                employee.setYears(rs.getInt("years"));
                                employee.setTotalDays(rs.getInt("
                                    totaldays"));
                                employee.setTakenDays(rs.getInt("
                                    takendays"));
                                employee.setHolidayStatus(
                                    HolidayStatusEnum.valueOf(rs.
```

```
                                        getString("status")));

                                employees.add(employee);
                        }

                        return employees;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public List<EmployeeModel> getEmployeeModelsOnHolidaysByDate(
    LocalDateTime date) {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT empl.id, empl.firstname,
                    empl.lastname, " +
                                "empl.role, empl.department, empl.
                                    accountid, hd.totaldays, " +
                                "hd.takendays, hd.status, empl.years,
                                    acc.email " +
                                "FROM employee empl " +
                                "inner join account acc " +
                                "on empl.accountid = acc.id " +
                                "inner join holiday_details hd " +
                                "on hd.employeeid = empl.id " +
                                "inner join holiday_request hr " +
                                "on hr.employeeid = empl.id " +
                                "WHERE hr.startdate <= ? AND hr.
                                    enddate >= ?;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, date);
                        stmt.setObject(2, date);
                        List<EmployeeModel> employees = new ArrayList
                            <>();
                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                EmployeeModel employee = new
                                    EmployeeModel();
                                employee.setId(UUID.fromString(rs.
                                    getString("id")));
                                employee.setFirstName(rs.getString("
                                    firstname"));
                                employee.setLastName(rs.getString("
                                    lastname"));
                                employee.setRole(EmployeeRoleEnum.
                                    valueOf(rs.getString("role")));
                                employee.setDepartment(DepartmentEnum.
```

```
                                        valueOf(rs.getString("department"))
                                        );
                                employee.setAccountId(UUID.fromString(
                                    rs.getString("accountid")));
                                employee.setEmail(rs.getString("email
                                    "));
                                employee.setYears(rs.getInt("years"));
                                employee.setTotalDays(rs.getInt("
                                    totaldays"));
                                employee.setTakenDays(rs.getInt("
                                    takendays"));
                                employee.setHolidayStatus(
                                    HolidayStatusEnum.valueOf(rs.
                                    getString("status")));

                                employees.add(employee);
                        }

                        return employees;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public List<EmployeeModel> getEmployeeModelsByDateAndHolidayStatus(
    LocalDateTime date, HolidayStatusEnum holidayStatus) {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT empl.id, empl.firstname,
                    empl.lastname, " +
                                "empl.role, empl.department, empl.
                                    accountid, hd.totaldays, " +
                                "hd.takendays, hd.status, empl.years,
                                    acc.email " +
                                "FROM employee empl " +
                                "inner join account acc " +
                                "on empl.accountid = acc.id " +
                                "inner join holiday_details hd " +
                                "on hd.employeeid = empl.id " +
                                "inner join holiday_request hr " +
                                "on hr.employeeid = empl.id " +
                                "WHERE hr.startdate <= ? AND hr.
                                    enddate >= ? AND "
                                + "hd.status = ?;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, date);
                        stmt.setObject(2, date);
                        stmt.setObject(3, holidayStatus.name());
                        List<EmployeeModel> employees = new ArrayList
```

```
                                        <>();
                                ResultSet rs = stmt.executeQuery();

                                while(rs.next()) {
                                        EmployeeModel employee = new
                                            EmployeeModel();
                                        employee.setId(UUID.fromString(rs.
                                            getString("id")));
                                        employee.setFirstName(rs.getString("
                                            firstname"));
                                        employee.setLastName(rs.getString("
                                            lastname"));
                                        employee.setRole(EmployeeRoleEnum.
                                            valueOf(rs.getString("role")));
                                        employee.setDepartment(DepartmentEnum.
                                            valueOf(rs.getString("department"))
                                            );
                                        employee.setAccountId(UUID.fromString(
                                            rs.getString("accountid")));
                                        employee.setEmail(rs.getString("email
                                            "));
                                        employee.setYears(rs.getInt("years"));
                                        employee.setTotalDays(rs.getInt("
                                            totaldays"));
                                        employee.setTakenDays(rs.getInt("
                                            takendays"));
                                        employee.setHolidayStatus(
                                            HolidayStatusEnum.valueOf(rs.
                                            getString("status")));

                                        employees.add(employee);
                                }

                                return employees;
                        }
                } catch(Exception ex) {
                        logger.error(ex.getMessage(), ex);
                }

                return null;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
```

```
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;

import javax.annotation.Resource;
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.sql.DataSource;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.HolidayDetailsEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.model.HolidayRequestModel;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;

/**
 * HolidayDetails Repository implementation using java.sql.PreparedStatement
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class HolidayDetailsRepository implements IHolidayDetailsRepository {

        private static final Logger logger = Logger.getLogger(
           HolidayDetailsRepository.class);

        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    DataSource dataSource;

        @Override
        public HolidayDetailsEntity findById(UUID id) {
                try (Connection connection = dataSource.getConnection()) {

                        final String query = "SELECT id, employeeid, status,
                           totaldays, takendays, created, modified "
                                        + "FROM holiday_details "
                                        + "WHERE id = ?";
```

```
                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, id);

                        ResultSet rs = stmt.executeQuery();
                        while(rs.next()) {
                                HolidayDetailsEntity entity = new
                                    HolidayDetailsEntity();
                                entity.setId(UUID.fromString(rs.
                                    getString("id")));
                                entity.setEmployeeId(UUID.fromString(
                                    rs.getString("employeeid")));
                                entity.setTotalDays(rs.getInt("
                                    totaldays"));
                                entity.setTakenDays(rs.getInt("
                                    takendays"));
                                entity.setStatus(rs.getString("status
                                    "));
                                entity.setCreated(LocalDateTime.parse(
                                    rs.getString("created"), DateUtils.
                                    FORMATTER));
                                entity.setModified(LocalDateTime.parse
                                    (rs.getString("modified"),
                                    DateUtils.FORMATTER));
                                return entity;
                        }

                        return null;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public HolidayDetailsEntity findByEmployeeId(UUID employeeId) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT id, employeeid, status,
                    totaldays, takendays, created, modified "
                                + "FROM holiday_details "
                                + "WHERE employeeId = ?";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, employeeId);

                        ResultSet rs = stmt.executeQuery();
                        while(rs.next()) {
                                HolidayDetailsEntity entity = new
                                    HolidayDetailsEntity();
```

```
                                entity.setId(UUID.fromString(rs.
                                    getString("id")));
                                entity.setEmployeeId(UUID.fromString(
                                    rs.getString("employeeid")));
                                entity.setTotalDays(rs.getInt("
                                    totaldays"));
                                entity.setTakenDays(rs.getInt("
                                    takendays"));
                                entity.setStatus(rs.getString("status
                                    "));
                                entity.setCreated(LocalDateTime.parse(
                                    rs.getString("created"), DateUtils.
                                    FORMATTER));
                                entity.setModified(LocalDateTime.parse
                                    (rs.getString("modified"),
                                    DateUtils.FORMATTER));

                                return entity;
                        }

                        return null;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public boolean save(HolidayDetailsEntity entity) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "INSERT INTO holiday_details (id,
                    employeeid, totaldays, takendays, "
                                + " status, created, modified) "
                                + "VALUES (?,?,?,?,?,?,?);";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, entity.getId());
                        stmt.setObject(2, entity.getEmployeeId());
                        stmt.setObject(3, entity.getTotalDays());
                        stmt.setObject(4, entity.getTakenDays());
                        stmt.setObject(5, entity.getStatus());
                        stmt.setObject(6, entity.getCreated());
                        stmt.setObject(7, entity.getModified());

                        if (stmt.executeUpdate() == 1) {
                                return true;
                        } else {
                                return false;
                        }
                }
```

```
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return false;
}

@Override
public List<HolidayDetailsEntity> getHolidayDetails() {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT id, employeeid, totaldays
                    , takendays, status, created, modified "
                                + " FROM holiday_details;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        List<HolidayDetailsEntity> entities = new
                            ArrayList<>();
                        ResultSet rs = stmt.executeQuery();
                        while(rs.next()) {
                                HolidayDetailsEntity entity = new
                                    HolidayDetailsEntity();

                                entity.setId(UUID.fromString(rs.
                                    getString("id")));
                                entity.setEmployeeId(UUID.fromString(
                                    rs.getString("employeeid")));
                                entity.setTotalDays(rs.getInt("
                                    totaldays"));
                                entity.setTakenDays(rs.getInt("
                                    takendays"));
                                entity.setStatus(rs.getString("status
                                    "));
                                entity.setCreated(LocalDateTime.parse(
                                    rs.getString("created"), DateUtils.
                                    FORMATTER));
                                entity.setModified(LocalDateTime.parse
                                    (rs.getString("modified"),
                                    DateUtils.FORMATTER));

                                entities.add(entity);
                        }

                        return entities;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
```

```
public List<HolidayRequestModel> getApprovedAndByDate(LocalDateTime
    date, HolidayRequestStatusEnum requestStatus) {
        try (Connection connection = dataSource.getConnection()) {
                final String query = "SELECT hr.id, hd.employeeid, hd.
                    takendays, hd.totaldays, " +
                                    "hr.startdate, hr.enddate, hd.
                                        status as holidaystatus, "
                                    + "hr.status as requeststatus
                                        " +
                                "FROM holiday_request hr " +
                                    "inner join holiday_details hd
                                        " +
                                    "on hd.employeeid = hr.
                                        employeeid " +
                                "WHERE hr.startdate <= ? AND hr.
                                    enddate >= ? AND " +
                                    "hr.status = ?;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, date);
                        stmt.setObject(2, date);
                        stmt.setObject(3, requestStatus.name());
                        List<HolidayRequestModel> requestModels = new
                            ArrayList<>();
                        ResultSet rs = stmt.executeQuery();

                        while(rs.next()) {
                                HolidayRequestModel model = new
                                    HolidayRequestModel();
                                model.setId(UUID.fromString(rs.
                                    getString("id")));
                                model.setEmployeeId(UUID.fromString(rs
                                    .getString("employeeid")));
                                model.setTotalDays(rs.getInt("
                                    totaldays"));
                                model.setTakenDays(rs.getInt("
                                    takendays"));
                                model.setHolidayStatus(
                                    HolidayStatusEnum.valueOf(rs.
                                    getString("holidaystatus")));
                                model.setRequestStatus(
                                    HolidayRequestStatusEnum.valueOf(rs
                                    .getString("requeststatus")));
                                model.setStartDate(LocalDateTime.parse
                                    (rs.getString("startdate"),
                                    DateUtils.FORMATTER));
                                model.setEndDate(LocalDateTime.parse(
                                    rs.getString("enddate"), DateUtils.
                                    FORMATTER));

                                requestModels.add(model);
                        }
```

```
                        return requestModels;
                    }
                } catch(Exception ex) {
                    logger.error(ex.getMessage(), ex);
                }

                return null;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;

import javax.annotation.Resource;
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.sql.DataSource;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.HolidayRequestEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.model.HolidayRequestModel;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;

/**
 * HolidayRequest Repository implementation using java.sql.PreparedStatement
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
```

```
 *
 */
@ApplicationScoped
@Default
public class HolidayRequestRepository implements IHolidayRequestRepository {

        private static final Logger logger = Logger.getLogger(
            HolidayRequestRepository.class);

        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    private DataSource dataSource;

        @Override
        public HolidayRequestEntity findById(UUID holidayRequestId) {
                try (Connection connection = dataSource.getConnection()) {

                        final String query = "SELECT id, employeeid, status,
                            startdate, enddate, created, modified "
                                        + "FROM holiday_request "
                                        + "WHERE id = ?";

                        try (PreparedStatement stmt = connection.
                            prepareStatement(query)) {
                                stmt.setObject(1, holidayRequestId);

                                HolidayRequestEntity entity = new
                                    HolidayRequestEntity();
                                ResultSet rs = stmt.executeQuery();
                                while(rs.next()) {
                                        entity.setId(UUID.fromString(rs.
                                            getString("id")));
                                        entity.setEmployeeId(UUID.fromString(
                                            rs.getString("employeeid")));
                                        entity.setStatus(rs.getString("status
                                            "));
                                        entity.setStartDate(LocalDateTime.
                                            parse(rs.getString("startdate"),
                                            DateUtils.FORMATTER));
                                        entity.setEndDate(LocalDateTime.parse(
                                            rs.getString("enddate"), DateUtils.
                                            FORMATTER));
                                        entity.setCreated(LocalDateTime.parse(
                                            rs.getString("created"), DateUtils.
                                            FORMATTER));
                                        entity.setModified(LocalDateTime.parse
                                            (rs.getString("modified"),
                                            DateUtils.FORMATTER));
                                }

                                return entity;
                        }
                } catch(Exception ex) {
                        logger.error(ex.getMessage(), ex);
                }
```

```
        return null;
}


@Override
public boolean save(HolidayRequestEntity holidayRequestEntity) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "INSERT INTO holiday_request (id,
                    employeeid, status, startdate, enddate, created,
                    modified) "
                                + "VALUES (?,?,?,?,?,?,?);";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, holidayRequestEntity.getId()
                            );
                        stmt.setObject(2, holidayRequestEntity.
                            getEmployeeId());
                        stmt.setString(3, holidayRequestEntity.
                            getStatus());
                        stmt.setObject(4, holidayRequestEntity.
                            getStartDate());
                        stmt.setObject(5, holidayRequestEntity.
                            getEndDate());
                        stmt.setObject(6, holidayRequestEntity.
                            getCreated());
                        stmt.setObject(7, holidayRequestEntity.
                            getModified());

                        if (stmt.executeUpdate() == 1) {
                            return true;
                        } else {
                            return false;
                        }
                }

        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return false;
}


@Override
public List<HolidayRequestEntity> getHolidayRequests() {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT id, employeeid, status,
                    startdate, enddate, created, modified "
                                + "FROM holiday_request";
```

```
                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        List<HolidayRequestEntity> holidayRequests =
                            new ArrayList<>();
                        ResultSet rs = stmt.executeQuery();
                        while(rs.next()) {
                                HolidayRequestEntity entity = new
                                    HolidayRequestEntity();
                                entity.setId(UUID.fromString(rs.
                                    getString("id")));
                                entity.setEmployeeId(UUID.fromString(
                                    rs.getString("employeeid")));
                                entity.setStatus(rs.getString("status
                                    "));
                                entity.setStartDate(LocalDateTime.
                                    parse(rs.getString("startdate"),
                                    DateUtils.FORMATTER));
                                entity.setEndDate(LocalDateTime.parse(
                                    rs.getString("enddate"), DateUtils.
                                    FORMATTER));
                                entity.setCreated(LocalDateTime.parse(
                                    rs.getString("created"), DateUtils.
                                    FORMATTER));
                                entity.setModified(LocalDateTime.parse
                                    (rs.getString("modified"),
                                    DateUtils.FORMATTER));

                                holidayRequests.add(entity);
                        }

                        return holidayRequests;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}


@Override
public HolidayRequestEntity update(UUID id, HolidayRequestEntity
    holidayRequestEntity) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "UPDATE holiday_request SET
                    status = ?, startdate = ?, enddate = ?, modified =
                    ?, employeeid = ? "
                                + "WHERE id= ?;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setString(1, holidayRequestEntity.
                            getStatus());
```

```
                        stmt.setObject(2, holidayRequestEntity.
                            getStartDate());
                        stmt.setObject(3, holidayRequestEntity.
                            getEndDate());
                        stmt.setObject(4, holidayRequestEntity.
                            getModified());
                        stmt.setObject(5, holidayRequestEntity.
                            getEmployeeId());
                        stmt.setObject(6, id);

                        if (stmt.executeUpdate() == 1) {
                            return holidayRequestEntity;
                        } else {
                            return null;
                        }
                    }
            } catch(Exception ex) {
                    logger.error(ex.getMessage(), ex);
            }

            return null;
    }


    @Override
    public List<HolidayRequestEntity> getHolidayRequestsByStatus(
        HolidayRequestStatusEnum requestStatus) {
            try (Connection connection = dataSource.getConnection()) {

                    final String query = "SELECT id, employeeid, status,
                        startdate, enddate, created, modified "
                                + "FROM holiday_request "
                                + "WHERE status = ?";

                    try (PreparedStatement stmt = connection.
                        prepareStatement(query)) {
                            stmt.setObject(1, requestStatus.name());

                            List<HolidayRequestEntity> holidayRequests =
                                new ArrayList<>();
                            ResultSet rs = stmt.executeQuery();
                            while(rs.next()) {
                                    HolidayRequestEntity entity = new
                                        HolidayRequestEntity();
                                    entity.setId(UUID.fromString(rs.
                                        getString("id")));
                                    entity.setEmployeeId(UUID.fromString(
                                        rs.getString("employeeid")));
                                    entity.setStatus(rs.getString("status
                                        "));
                                    entity.setStartDate(LocalDateTime.
                                        parse(rs.getString("startdate"),
                                        DateUtils.FORMATTER));
                                    entity.setEndDate(LocalDateTime.parse(
```

```
                                        rs.getString("enddate"), DateUtils.
                                            FORMATTER));
                                    entity.setCreated(LocalDateTime.parse(
                                        rs.getString("created"), DateUtils.
                                            FORMATTER));
                                    entity.setModified(LocalDateTime.parse
                                        (rs.getString("modified"),
                                        DateUtils.FORMATTER));

                                    holidayRequests.add(entity);
                            }

                            return holidayRequests;
                    }
            } catch(Exception ex) {
                    logger.error(ex.getMessage(), ex);
            }

            return null;
    }

    @Override
    public List<HolidayRequestModel> getHolidayRequestModels() {
            try (Connection connection = dataSource.getConnection()) {
                    final String query = "SELECT hr.id, hr.employeeid, hr.
                        status, " +
                                    "hr.startdate, hr.enddate, " +
                                    "hd.totaldays, hd.takendays, empl.
                                        years " +
                                    "FROM holiday_request hr " +
                                    "inner join employee empl " +
                                    "on empl.id = hr.employeeid " +
                                    "inner join holiday_details hd " +
                                    "on hd.employeeid = empl.id "
                                    + "ORDER BY hr.created "
                                    + "limit 100;";

                    try (PreparedStatement stmt = connection.
                        prepareStatement(query)) {

                            List<HolidayRequestModel> models = new
                                ArrayList<>();
                            ResultSet rs = stmt.executeQuery();


                            while(rs.next()) {

                                    HolidayRequestModel model = new
                                        HolidayRequestModel();
                                    model.setId(UUID.fromString(rs.
                                        getString("id")));
                                    model.setEmployeeId(UUID.fromString(rs
                                        .getString("employeeid")));
                                    model.setRequestStatus(
```

```
                                        HolidayRequestStatusEnum.valueOf(rs
                                            .getString("status")));
                                    model.setStartDate(LocalDateTime.parse
                                        (rs.getString("startdate"),
                                        DateUtils.FORMATTER));
                                    model.setEndDate(LocalDateTime.parse(
                                        rs.getString("enddate"), DateUtils.
                                        FORMATTER));
                                    model.setYears(rs.getInt("years"));
                                    model.setTotalDays(rs.getInt("
                                        totaldays"));
                                    model.setTakenDays(rs.getInt("
                                        takendays"));

                                    models.add(model);
                                }

                                return models;
                            }
                    } catch(Exception ex) {
                            logger.error(ex.getMessage(), ex);
                    }

                    return null;
            }

            @Override
            public List<HolidayRequestModel> getHolidayRequestModels(int offset,
                int limit) {
                    try (Connection connection = dataSource.getConnection()) {
                            final String query = "SELECT hr.id, hr.employeeid, hr.
                                status, " +
                                            "hr.startdate, hr.enddate, " +
                                            "hd.totaldays, hd.takendays, empl.
                                                years " +
                                            "FROM holiday_request hr " +
                                            "inner join employee empl " +
                                            "on empl.id = hr.employeeid " +
                                            "inner join holiday_details hd " +
                                            "on hd.employeeid = empl.id "
                                            + "ORDER BY hr.created "
                                            + "limit ? offset ?;";

                            try (PreparedStatement stmt = connection.
                                prepareStatement(query)) {
                                    stmt.setInt(1, limit);
                                    stmt.setInt(2, offset);

                                    List<HolidayRequestModel> models = new
                                        ArrayList<>();
                                    ResultSet rs = stmt.executeQuery();


                                    while(rs.next()) {
```

```
                                HolidayRequestModel model = new
                                    HolidayRequestModel();
                                model.setId(UUID.fromString(rs.
                                    getString("id")));
                                model.setEmployeeId(UUID.fromString(rs
                                    .getString("employeeid")));
                                model.setRequestStatus(
                                    HolidayRequestStatusEnum.valueOf(rs
                                    .getString("status")));
                                model.setStartDate(LocalDateTime.parse
                                    (rs.getString("startdate"),
                                    DateUtils.FORMATTER));
                                model.setEndDate(LocalDateTime.parse(
                                    rs.getString("enddate"), DateUtils.
                                    FORMATTER));
                                model.setYears(rs.getInt("years"));
                                model.setTotalDays(rs.getInt("
                                    totaldays"));
                                model.setTakenDays(rs.getInt("
                                    takendays"));

                                models.add(model);
                        }

                        return models;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}

@Override
public List<HolidayRequestModel> getHolidayRequestModelsByStatus(
    HolidayRequestStatusEnum requestStatus) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT hr.id, hr.employeeid, hr.
                    status, " +
                                "hr.startdate, hr.enddate, " +
                                "hd.totaldays, hd.takendays, empl.
                                    years " +
                                "FROM holiday_request hr " +
                                "inner join employee empl " +
                                "on empl.id = hr.employeeid " +
                                "inner join holiday_details hd " +
                                "on hd.employeeid = empl.id " +
                                "WHERE hr.status = ?;";

                try (PreparedStatement stmt = connection.
                    prepareStatement(query)) {
                        stmt.setObject(1, requestStatus.name());
```

```
                                List<HolidayRequestModel> models = new
                                    ArrayList<>();
                                ResultSet rs = stmt.executeQuery();

                                while(rs.next()) {

                                        HolidayRequestModel model = new
                                            HolidayRequestModel();
                                        model.setId(UUID.fromString(rs.
                                            getString("id")));
                                        model.setEmployeeId(UUID.fromString(rs
                                            .getString("employeeid")));
                                        model.setRequestStatus(
                                            HolidayRequestStatusEnum.valueOf(rs
                                            .getString("status")));
                                        model.setStartDate(LocalDateTime.parse
                                            (rs.getString("startdate"),
                                            DateUtils.FORMATTER));
                                        model.setEndDate(LocalDateTime.parse(
                                            rs.getString("enddate"), DateUtils.
                                            FORMATTER));
                                        model.setYears(rs.getInt("years"));
                                        model.setTotalDays(rs.getInt("
                                            totaldays"));
                                        model.setTakenDays(rs.getInt("
                                            takendays"));

                                        models.add(model);
                                }

                                return models;
                        }
                } catch(Exception ex) {
                        logger.error(ex.getMessage(), ex);
                }

                return null;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
```

```
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;

import com.holidaysystem.entity.AccountEntity;

/**
 * Interface for Account repository, which can be supported by
 * all implementations of the interface
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IAccountRepository {
        /**
         * Finds account by account id
         * @param accountId UUID account id
         * @return account entity
         */
        AccountEntity findById(UUID accountId);
    /**
     * Finds account by email
     * @param email account email
     * @return account entity
     */
        AccountEntity findByEmail(String email);
        /**
         * Finds account by email and password
         * @param email account email
         * @param password account password
         * @return account entity
         */
        AccountEntity findByEmailAndPassword(String email, String password);
    /**
     * Adds a new account into persistence storage
     * @param account new account entity
     * @return saved account entity
     */
    boolean save(AccountEntity account);
    /**
     * Generated hashed password of the password
     * @param password account password
     * @return hashed password
     */
    String generateHashedPassword(String password);
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
```

```
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.List;
import java.util.UUID;

import com.holidaysystem.entity.AuthorizationRoleEntity;

/**
 * Interface for AuthorizationRole repository, which can be supported by
 * all implementations of the interface
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IAuthorizationRoleRepository {
        /**
         * Gets all authorization roles
         * @return list of AuthorizationRoleEntity
         */
        List<AuthorizationRoleEntity> getAll();
        /**
         * Gets authorization role by id
         * @param id authorization role id
         * @return AuthorizationRoleEntity
         */
        AuthorizationRoleEntity findById(UUID id);
        /**
         * Gets authorization role by name
         * @param name authorization role name
         * @return AuthorizationRoleEntity
         */
        AuthorizationRoleEntity findByName(String name);
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
```

```
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;
import java.time.LocalDateTime;
import java.util.List;
import com.holidaysystem.entity.EmployeeEntity;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.model.EmployeeModel;

/**
 * Interface for Employee repository, which can be supported by
 * all implementations of the interface
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IEmployeeRepository {
        /**
         * Finds employee entity by id
         * @param employeeId UUID employee id
         * @return employee entity
         */
        EmployeeEntity findById(UUID employeeId);
    /**
     * Finds employee entity by email
     * @param email
     * @return employee entity
     */
    EmployeeEntity findByEmail(String email);
    /**
     * Adds a new employee
     * @param employeeEntity employee entity
     * @return true is added, false is failed
     */
    boolean save(EmployeeEntity employeeEntity);
    /**
     * Gets all employee entities
     * @return list of employee entities
     */
    List<EmployeeEntity> getEmployees();
    /**
     * Gets all employee models
     * @return list of employee models
     */
    List<EmployeeModel> getEmployeeModels();
    /**
```

```
     * Gets employees from specific department
     * @param department department
     * @return list of employee models
     */
    List<EmployeeModel> getEmployeeModelsByDepartmentId(String department);
    /**
     * Gets employees on holidays on specific date
     * @param date the date
     * @return list of employee models on holidays
     */
    List<EmployeeModel> getEmployeeModelsOnHolidaysByDate(LocalDateTime date);
    /**
     * Gets employees on holidays or on duty on specific date
     * @param date the date
     * @param holidayStatus the status of HolidayStatusEnum
     * @return list of employee models
     */
    List<EmployeeModel> getEmployeeModelsByDateAndHolidayStatus(LocalDateTime
        date, HolidayStatusEnum holidayStatus);
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;
import java.time.LocalDateTime;
import java.util.List;
import com.holidaysystem.entity.HolidayDetailsEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.model.HolidayRequestModel;

/**
 * Interface for Holiday Details repository, which can be supported by
 * all implementations of the interface
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IHolidayDetailsRepository {
```

```
        /**
         * Finds holiday details entity by id
         * @param Id UUID of holiday details entity
         * @return HolidayDetailsEntity entity
         */
        HolidayDetailsEntity findById(UUID Id);
        /**
         * Finds employee by employee Id
         * @param employeeId
         * @return HolidayDetailsEntity entity
         */
        HolidayDetailsEntity findByEmployeeId(UUID employeeId);
    /**
     * Adds a new holiday detail entity
     * @param holidayDetails
     * @return true is saved, false is failed
     */
    boolean save(HolidayDetailsEntity holidayDetails);
    /**
     * Fetches all holiday detail entities
     * @return list of HolidayDetailsEntity entities
     */
    List<HolidayDetailsEntity> getHolidayDetails();
    /**
     * Fetches HolidayRequest models
     * @param date LocalDateTime
     * @param requestStatus HolidayRequestStatusEnum enumeration
     * @return list of HolidayRequestModel models
     */
    List<HolidayRequestModel> getApprovedAndByDate(LocalDateTime date,
        HolidayRequestStatusEnum requestStatus);
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;
import java.util.List;
```

```
import com.holidaysystem.entity.HolidayRequestEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.model.HolidayRequestModel;

/**
 * Interface for Holiday Request repository, which can be supported by
 * all implementations of the interface
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IHolidayRequestRepository {
        List<HolidayRequestEntity> getHolidayRequests();
        List<HolidayRequestModel> getHolidayRequestModels();
        List<HolidayRequestModel> getHolidayRequestModels(int offset, int
            limit);
        List<HolidayRequestEntity> getHolidayRequestsByStatus(
            HolidayRequestStatusEnum requestStatus);
        List<HolidayRequestModel> getHolidayRequestModelsByStatus(
            HolidayRequestStatusEnum requestStatus);
        HolidayRequestEntity findById(UUID holidayRequestId);
        boolean save(HolidayRequestEntity holidayRequestEntity);
        HolidayRequestEntity update(UUID id, HolidayRequestEntity
            holidayRequestEntity);
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;

import java.util.UUID;

import javax.annotation.Resource;
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.sql.DataSource;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
```

```
import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.RequestAlertEntity;

import jakarta.inject.Named;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;

/**
 * RequestAlert Repository implementation using java.sql.PreparedStatement
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class RequestAlertRepository implements IRequestAlertRepository {

        private static final Logger logger = Logger.getLogger(
           RequestAlertRepository.class);

        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    private DataSource dataSource;

        @Override
        public RequestAlertEntity findById(UUID requestAlertId) {
                try (Connection connection = dataSource.getConnection()) {

                        final String query = "SELECT id, requestid, date,
                           created, modified FROM request_alert_queue WHERE id
                            = ?";

                        try (PreparedStatement stmt = connection.
                           prepareStatement(query)) {
                                stmt.setObject(1, requestAlertId);

                                RequestAlertEntity entity = new
                                   RequestAlertEntity();
                                ResultSet rs = stmt.executeQuery();
                                while(rs.next()) {
                                        entity.setId(UUID.fromString(rs.
                                           getString("id")));
                                        entity.setRequestId(UUID.fromString(rs
                                           .getString("requestid")));
                                        entity.setDate(LocalDateTime.parse(rs.
                                           getString("startdate"), DateUtils.
                                           FORMATTER));
                                        entity.setCreated(LocalDateTime.parse(
                                           rs.getString("created"), DateUtils.
                                           FORMATTER));
                                        entity.setModified(LocalDateTime.parse
```

```
                                        (rs.getString("modified"),
                                        DateUtils.FORMATTER));
                        }

                        return entity;
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return null;
}


@Override
public boolean save(RequestAlertEntity requestAlertEntity) {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "INSERT INTO request_alert_queue
                   (id, requestid, date, created, modified) "
                                + "VALUES (?,?,?,?,?);";

                try (PreparedStatement stmt = connection.
                   prepareStatement(query)) {
                        stmt.setObject(1, requestAlertEntity.getId());
                        stmt.setObject(2, requestAlertEntity.
                           getRequestId());
                        stmt.setObject(3, requestAlertEntity.getDate()
                           );
                        stmt.setObject(4, requestAlertEntity.
                           getCreated());
                        stmt.setObject(5, requestAlertEntity.
                           getModified());

                        if (stmt.executeUpdate() == 1) {
                                return true;
                        } else {
                                return false;
                        }
                }
        } catch(Exception ex) {
                logger.error(ex.getMessage(), ex);
        }

        return false;
}


@Override
public List<RequestAlertEntity> getRequestAlerts() {
        try (Connection connection = dataSource.getConnection()) {

                final String query = "SELECT id, requestid, date,
                   created, modified FROM request_alert_queue";
```

```
                        try (PreparedStatement stmt = connection.
                            prepareStatement(query)) {
                                List<RequestAlertEntity> requestAlerts = new
                                    ArrayList<>();
                                ResultSet rs = stmt.executeQuery();
                                while(rs.next()) {
                                        RequestAlertEntity entity = new
                                            RequestAlertEntity();
                                        entity.setId(UUID.fromString(rs.
                                            getString("id")));
                                        entity.setRequestId(UUID.fromString(rs
                                            .getString("requestid")));
                                        entity.setDate(LocalDateTime.parse(rs.
                                            getString("date"), DateUtils.
                                            FORMATTER));
                                        entity.setCreated(LocalDateTime.parse(
                                            rs.getString("created"), DateUtils.
                                            FORMATTER));
                                        entity.setModified(LocalDateTime.parse
                                            (rs.getString("modified"),
                                            DateUtils.FORMATTER));

                                        requestAlerts.add(entity);
                                }

                                return requestAlerts;
                        }
                } catch(Exception ex) {
                        logger.error(ex.getMessage(), ex);
                }

                return null;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.repository;
```

```
import java.util.UUID;
import java.util.List;
import com.holidaysystem.entity.RequestAlertEntity;

/**
 * Interface for Request Alert repository, which can be supported by
 * all implementations of the interface
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IRequestAlertRepository {
        List<RequestAlertEntity> getRequestAlerts();
        RequestAlertEntity findById(UUID holidayRequestId);
        boolean save(RequestAlertEntity holidayRequestEntity);
}



/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.ws.rs.Consumes;
import javax.ws.rs.POST;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;
import javax.ws.rs.core.Response.Status;

import org.jboss.logging.Logger;

import com.holidaysystem.mapper.AccountMapper;
import com.holidaysystem.model.AccountDetailsModel;
import com.holidaysystem.vo.AccountResponse;
import com.holidaysystem.vo.LoginRequest;
import com.holidaysystem.vo.RegistrationRequest;
import com.holidaysystem.service.IAuthService;
```

```
import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;

/**
 * REST API for auth resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/auth")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
public class AuthResource {

        private static final Logger logger = Logger.getLogger(AuthResource.
            class);

        @Inject
        private IAuthService authService;
        @Inject
        private AccountMapper accountMapper;

    @POST
    @Path("/register")
    @Consumes(MediaType.APPLICATION_JSON)
    public Response register(RegistrationRequest registrationRequest) {

        logger.debug("register");

        AccountDetailsModel accountModel = authService
                        .register(registrationRequest);

        AccountResponse resp = accountMapper.toResponse(accountModel);

        return Response.ok(resp)
                        .header("Access-Control-Allow-Origin", "*")
                        .status(Status.CREATED)
                        .build();
    }

    @POST
    @Path("/login")
    @Consumes(MediaType.APPLICATION_JSON)
    public Response login(LoginRequest loginRequest) {

        AccountDetailsModel accountModel = authService.login(loginRequest.
            getEmail(), loginRequest.getPassword());

        if(accountModel == null) {
                logger.info("account not found");

                return Response.noContent()
                                .header("Access-Control-Allow-Origin", "*")
                                .status(Status.NOT_FOUND)
```

```
                                .build();
        }

        AccountResponse resp = accountMapper.toResponse(accountModel);

        return Response.ok(resp)
                       .header("Access-Control-Allow-Origin", "*")
                       .build();
    }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;

import org.jboss.logging.Logger;

import java.util.ArrayList;
import java.util.List;

import com.holidaysystem.enumeration.DepartmentEnum;

/**
 * REST API for departments resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/department")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
```

```
//@ServletSecurity(value = @HttpConstraint(rolesAllowed = {"USER", "ADMIN"}))
public class DepartmentsResource {

        private static final Logger logger = Logger.getLogger(
           DepartmentsResource.class);

    @GET
    @Path("/all")
    public Response getDepartments() {
        logger.debug("getDepartments()");

        List<String> departments = new ArrayList<>();

        for(DepartmentEnum value: DepartmentEnum.values()) {
                departments.add(value.name());
        }

        return Response.ok(departments)
                       .build();
    }

}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.POST;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;
import javax.ws.rs.core.Response.Status;
```

```
import org.jboss.logging.Logger;

import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

import com.holidaysystem.common.DateUtils;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.mapper.EmployeeMapper;
import com.holidaysystem.model.EmployeeModel;
import com.holidaysystem.vo.NewEmployeeRequest;
import com.holidaysystem.vo.EmployeeResponse;
import com.holidaysystem.vo.FindEmployeesByDateRequest;
import com.holidaysystem.service.IEmployeeService;

/**
 * REST API for employee resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/employee")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
//@ServletSecurity(value = @HttpConstraint(rolesAllowed = {"admin_role"}))
public class EmployeeResource {

        private static final Logger logger = Logger.getLogger(EmployeeResource
            .class);

        @Inject
        private EmployeeMapper employeeMapper;
        @Inject
        private IEmployeeService employeeService;


    @GET
    @Path("/all")
    public Response getEmployees() {

        logger.debug("getEmployees()");

        List<EmployeeModel> empployeeModels = employeeService.getEmployees();
        List<EmployeeResponse> employees = new ArrayList<>();
        for(EmployeeModel empployeeModel: empployeeModels) {
                EmployeeResponse employee = employeeMapper.toResponse(
                    empployeeModel);
                employees.add(employee);
        }

        return Response.ok(employees)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
```

```
}

@GET()
@Path("/{id}")
public Response getEmployee(@PathParam("id") UUID id) {
    EmployeeModel empployeeModel = employeeService.findById(id);
    EmployeeResponse employee = employeeMapper.toResponse(empployeeModel);
    return Response.ok(employee)
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
}

@POST()
public Response createEmployee(NewEmployeeRequest employeeRequest) {

    EmployeeModel employeeModel = employeeService.create(employeeRequest);
    EmployeeResponse employee = employeeMapper.toResponse(employeeModel);

    return Response.status(Status.CREATED).entity(employee).build();
}

@POST()
@Path("all/date")
public Response findEmployees(FindEmployeesByDateRequest request) {

    String strDate = request.getDate();
    LocalDateTime date = LocalDateTime.parse(strDate, DateUtils.FORMATTER)
        ;

    List<EmployeeModel> employeeModels = employeeService.
        getEmployeesByDate(date);

    List<EmployeeResponse> employees = new ArrayList<>();
    for(EmployeeModel empployeeModel: employeeModels) {
            EmployeeResponse employee = employeeMapper.toResponse(
                empployeeModel);
            employees.add(employee);
    }

    return Response.ok(employees)
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
}

@POST()
@Path("on-duty/date")
public Response findEmployeesOnDuty(FindEmployeesByDateRequest request) {

    String strDate = request.getDate();
    LocalDateTime date = LocalDateTime.parse(strDate, DateUtils.FORMATTER)
        ;

    List<EmployeeModel> employeeModels = employeeService.
        getEmployeesByDateAndHolidayStatus(date, HolidayStatusEnum.ON_DUTY)
```

```
            ;

        List<EmployeeResponse> employees = new ArrayList<>();
        for(EmployeeModel empployeeModel: employeeModels) {
                EmployeeResponse employee = employeeMapper.toResponse(
                    empployeeModel);
                employees.add(employee);
        }

        return Response.ok(employees)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
    }

    @POST()
    @Path("on-holiday/date")
    public Response findEmployeesOnHoliday(FindEmployeesByDateRequest request)
        {

        String strDate = request.getDate();
        LocalDateTime date = LocalDateTime.parse(strDate, DateUtils.FORMATTER)
            ;

        List<EmployeeModel> employeeModels = employeeService.
            getEmployeesByDateAndHolidayStatus(date, HolidayStatusEnum.
            ON_HOLIDAY);

        List<EmployeeResponse> employees = new ArrayList<>();
        for(EmployeeModel empployeeModel: employeeModels) {
                EmployeeResponse employee = employeeMapper.toResponse(
                    empployeeModel);
                employees.add(employee);
        }

        return Response.ok(employees)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
```

```
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;

/**
 * REST API for checking the system is running
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/hello")
@RequestScoped
public class HelloResource {

        @GET
        @Produces(MediaType.APPLICATION_JSON)
        public Response get() {

                return Response.status(Response.Status.OK)
                                .entity("Vacation system is running")
                                .header("Access-Control-Allow-Origin", "*")
                                .build();
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;
import javax.ws.rs.Consumes;
```

```
import javax.ws.rs.GET;
import javax.ws.rs.POST;
import javax.ws.rs.PUT;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.QueryParam;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;
import javax.ws.rs.core.Response.Status;

import org.jboss.logging.Logger;

import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.HolidayRequestEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.mapper.HolidayRequestMapper;
import com.holidaysystem.model.HolidayRequestModel;
import com.holidaysystem.model.PrioritizedRequestModel;
import com.holidaysystem.vo.HolidayRequest;
import com.holidaysystem.vo.HolidayResponse;
import com.holidaysystem.vo.PrioritizedRecordsRequest;
import com.holidaysystem.service.IHolidayRequestService;

/**
 * REST API for holiday request resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/holiday-request")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
//@ServletSecurity(value = @HttpConstraint(rolesAllowed = {"admin_role"}))
public class HolidayRequestResource {

        private static final Logger logger = Logger.getLogger(
            HolidayRequestResource.class);

        @Inject
        private IHolidayRequestService holidayRequestService;
        @Inject
    private HolidayRequestMapper holidayRequestMapper;

        /**
         * Gets all holiday requests
         * limit 100
         * @return
         */
```

```
@GET
@Path("/all")
public Response getHolidayRequests() {

    List<HolidayRequestModel> models = holidayRequestService.
        getHolidayRequests();

    List<HolidayResponse> holidayRequestResponses = new ArrayList<>();
    for(HolidayRequestModel model: models) {
            HolidayResponse holidayResponse = holidayRequestMapper.
                toResponse(model);
            holidayRequestResponses.add(holidayResponse);
    }

    return Response.ok(holidayRequestResponses)
                    .build();
}

/**
 * Gets pages of holiday requests sorted by created by default
 * @param offset skip number of requests
 * @param limit size of the page
 * @return list of HolidayResponse
 */
@GET
@Path("/query")
public Response getHolidayRequestPages(@QueryParam("offset") int offset,
            @QueryParam("limit") int limit) {

    List<HolidayRequestModel> models = holidayRequestService.
        getHolidayRequests(offset, limit);

    List<HolidayResponse> holidayRequestResponses = new ArrayList<>();
    for(HolidayRequestModel model: models) {
            HolidayResponse holidayResponse = holidayRequestMapper.
                toResponse(model);
            holidayRequestResponses.add(holidayResponse);
    }

    return Response.ok(holidayRequestResponses)
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
}

@GET
@Path("/all/{status}")
public Response getHolidayRequests(@PathParam("status")
    HolidayRequestStatusEnum status) {

    logger.info("getHolidayRequests() with status " + status.name());

    List<HolidayRequestModel> models = holidayRequestService.
        getHolidayRequestsByStatus(status);
```

```
    List<HolidayResponse> holidayRequestResponses = new ArrayList<>();

    for(HolidayRequestModel model: models) {
            HolidayResponse holidayResponse = holidayRequestMapper.
                toResponse(model);
            holidayRequestResponses.add(holidayResponse);
    }

    return Response.ok(holidayRequestResponses)
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
}

@GET
@Path("/all/prioritized")
public Response getPrioritizedHolidayRequests(PrioritizedRecordsRequest
    request) {

    LocalDateTime date = LocalDateTime.parse(request.getDate(), DateUtils.
        FORMATTER);
    HolidayRequestStatusEnum requestStatus = HolidayRequestStatusEnum.
        valueOf(request.getRequestStatus());

    List<PrioritizedRequestModel> models = holidayRequestService
                    .getPrioritizedHolidayRequests(date, requestStatus);

    List<UUID> requestIds = new ArrayList<>();

    for(PrioritizedRequestModel model: models) {
            requestIds.add(model.getRequestId());
    }

    return Response.ok(requestIds)
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
}

@GET
@Path("/{id}/alternative-dates")
public Response getAlternativeDates(@PathParam("id") UUID id) {

    //holidayRequestService.getAlternativeDates(id);

    return Response.ok()
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
}

@GET()
@Path("/{id}")
public Response getHolidayRequestById(@PathParam("id") UUID id) {

    HolidayRequestModel model = holidayRequestService.fetchModelById(id);
```

```
        HolidayResponse holidayRequestResp = holidayRequestMapper.toResponse(
            model);

        return Response.ok(holidayRequestResp)
                       .header("Access-Control-Allow-Origin", "*")
                       .build();
    }

    @POST()
    @Consumes(MediaType.APPLICATION_JSON)
    public Response createHolidayRequest(HolidayRequest holidayRequest) {
        UUID id = UUID.randomUUID();

        holidayRequestService.addHolidayRequest(id, holidayRequest);

        return Response.ok(id)
                       //.header("Access-Control-Allow-Origin", "*")
                       .status(Status.CREATED)
                       .build();
    }

    @PUT()
    @Path("/{id}")
    public Response updateHolidayRequest(@PathParam("id") UUID id,
        HolidayRequest holidayRequest) {
        HolidayRequestEntity updatedEntity = holidayRequestService.update(id,
            holidayRequest);

        if(updatedEntity != null) {
                HolidayResponse holidayRequestResp = holidayRequestMapper.
                    toResponse(updatedEntity);

                return Response.ok(holidayRequestResp)
                               .header("Access-Control-Allow-Origin", "*")
                               .build();
        }

        return Response.ok(null)
                       .header("Access-Control-Allow-Origin", "*")
                       .build();

    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
```

```
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;

import org.jboss.logging.Logger;

import java.util.ArrayList;
import java.util.List;

import com.holidaysystem.enumeration.EmployeeRoleEnum;

/**
 * REST API for roles resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/role")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
//@ServletSecurity(value = @HttpConstraint(rolesAllowed = {"admin_role"}))
public class RolesResource {

        private static final Logger logger = Logger.getLogger(RolesResource.
            class);

    @GET
    @Path("/all")
    public Response getRoles() {

        logger.debug("getRoles()");

        List<String> roles = new ArrayList<>();

        for(EmployeeRoleEnum value: EmployeeRoleEnum.values()) {
                roles.add(value.name());
        }

        return Response.ok(roles)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
```

```
    }

}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;

import org.jboss.logging.Logger;

import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

import com.holidaysystem.entity.RequestAlertEntity;
import com.holidaysystem.mapper.RequestAlertMapper;
import com.holidaysystem.vo.RequestAlertResponse;
import com.holidaysystem.repository.RequestAlertRepository;

/**
 * REST API for request-alert resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/request-alert")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
//@ServletSecurity(value = @HttpConstraint(rolesAllowed = {"admin_role"}))
```

```
public class RequestAlertResource {

        private static final Logger logger = Logger.getLogger(
            RequestAlertResource.class);

    @Inject
    private RequestAlertRepository requestAlertRepository;
    @Inject
    private RequestAlertMapper requestAlertMapper;

    @GET
    @Path("/all")
    public Response getRequestsAlerts() {
        logger.debug("getRequestsAlerts()");

        List<RequestAlertEntity> entities = requestAlertRepository.
            getRequestAlerts();
        List<RequestAlertResponse> requestAlertResponses = new ArrayList<>();
        for(RequestAlertEntity entity: entities) {
                RequestAlertResponse holidayResponse = requestAlertMapper.
                    toResponse(entity);
                requestAlertResponses.add(holidayResponse);
        }

        return Response.ok(requestAlertResponses)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
    }

    @GET()
    @Path("/{id}")
    public Response getHolidayRequestById(@PathParam("id") UUID id) {
        RequestAlertEntity entity = requestAlertRepository.findById(id);
        RequestAlertResponse requestAlertResp = requestAlertMapper.toResponse(
            entity);
        return Response.ok(requestAlertResp)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
```

```
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;

import org.jboss.logging.Logger;

import java.util.UUID;

import com.holidaysystem.vo.RequestValidationResponse;
import com.holidaysystem.service.IHolidayRequestService;

/**
 * REST API for holiday request resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/request-processing")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
//@ServletSecurity(value = @HttpConstraint(rolesAllowed = {"admin_role"}))
public class RequestProcessingResource {

        private static final Logger logger = Logger.getLogger(
            RequestProcessingResource.class);

    @Inject
    private IHolidayRequestService holidayRequestService;

    @GET()
    @Path("/{id}/validate")
    public Response validate(@PathParam("id") UUID holidayRequestId) {

        logger.debug("validate request id " + holidayRequestId);

        boolean valid = holidayRequestService.validate(holidayRequestId);

        RequestValidationResponse resp = new RequestValidationResponse();
        resp.setRequestId(holidayRequestId);
        resp.setValid(valid);

        return Response.ok(resp)
```

```
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
    }
}
```

] package com.holidaysystem.resource;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.ws.rs.Consumes;
import javax.ws.rs.POST;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;
import javax.ws.rs.core.Response.Status;

import org.jboss.logging.Logger;

import com.holidaysystem.mapper.AccountMapper;
import com.holidaysystem.model.AccountDetailsModel;
import com.holidaysystem.vo.AccountResponse;
import com.holidaysystem.vo.LoginRequest;
import com.holidaysystem.vo.RegistrationRequest;
import com.holidaysystem.service.IAuthService;

import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;

/**
 * REST API for auth resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/auth")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
```

```
public class AuthResource {

        private static final Logger logger = Logger.getLogger(AuthResource.
            class);

        @Inject
        private IAuthService authService;
        @Inject
        private AccountMapper accountMapper;

    @POST
    @Path("/register")
    @Consumes(MediaType.APPLICATION_JSON)
    public Response register(RegistrationRequest registrationRequest) {

        logger.debug("register");

        AccountDetailsModel accountModel = authService
                        .register(registrationRequest);

        AccountResponse resp = accountMapper.toResponse(accountModel);

        return Response.ok(resp)
                        .header("Access-Control-Allow-Origin", "*")
                        .status(Status.CREATED)
                        .build();
    }

    @POST
    @Path("/login")
    @Consumes(MediaType.APPLICATION_JSON)
    public Response login(LoginRequest loginRequest) {

        AccountDetailsModel accountModel = authService.login(loginRequest.
            getEmail(), loginRequest.getPassword());

        if(accountModel == null) {
                logger.info("account not found");

                return Response.noContent()
                                .header("Access-Control-Allow-Origin", "*")
                                .status(Status.NOT_FOUND)
                                .build();
        }

        AccountResponse resp = accountMapper.toResponse(accountModel);

        return Response.ok(resp)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
    }
}

/*
```

```
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;

import org.jboss.logging.Logger;

import java.util.ArrayList;
import java.util.List;

import com.holidaysystem.enumeration.DepartmentEnum;

/**
 * REST API for departments resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/department")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
//@ServletSecurity(value = @HttpConstraint(rolesAllowed = {"USER", "ADMIN"}))
public class DepartmentsResource {

        private static final Logger logger = Logger.getLogger(
            DepartmentsResource.class);

    @GET
    @Path("/all")
    public Response getDepartments() {
        logger.debug("getDepartments()");

        List<String> departments = new ArrayList<>();
```

```
        for(DepartmentEnum value: DepartmentEnum.values()) {
                departments.add(value.name());
        }

        return Response.ok(departments)
                        .build();
    }

}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.POST;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;
import javax.ws.rs.core.Response.Status;

import org.jboss.logging.Logger;

import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

import com.holidaysystem.common.DateUtils;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.mapper.EmployeeMapper;
import com.holidaysystem.model.EmployeeModel;
import com.holidaysystem.vo.NewEmployeeRequest;
```

```
import com.holidaysystem.vo.EmployeeResponse;
import com.holidaysystem.vo.FindEmployeesByDateRequest;
import com.holidaysystem.service.IEmployeeService;

/**
 * REST API for employee resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/employee")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
//@ServletSecurity(value = @HttpConstraint(rolesAllowed = {"admin_role"}))
public class EmployeeResource {

        private static final Logger logger = Logger.getLogger(EmployeeResource
            .class);

        @Inject
        private EmployeeMapper employeeMapper;
        @Inject
        private IEmployeeService employeeService;


    @GET
    @Path("/all")
    public Response getEmployees() {

        logger.debug("getEmployees()");

        List<EmployeeModel> empployeeModels = employeeService.getEmployees();
        List<EmployeeResponse> employees = new ArrayList<>();
        for(EmployeeModel empployeeModel: empployeeModels) {
                EmployeeResponse employee = employeeMapper.toResponse(
                    empployeeModel);
                employees.add(employee);
        }

        return Response.ok(employees)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
    }

    @GET()
    @Path("/{id}")
    public Response getEmployee(@PathParam("id") UUID id) {
        EmployeeModel empployeeModel = employeeService.findById(id);
        EmployeeResponse employee = employeeMapper.toResponse(empployeeModel);
        return Response.ok(employee)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
    }
```

```
@POST()
public Response createEmployee(NewEmployeeRequest employeeRequest) {

    EmployeeModel employeeModel = employeeService.create(employeeRequest);
    EmployeeResponse employee = employeeMapper.toResponse(employeeModel);

    return Response.status(Status.CREATED).entity(employee).build();
}

@POST()
@Path("all/date")
public Response findEmployees(FindEmployeesByDateRequest request) {

    String strDate = request.getDate();
    LocalDateTime date = LocalDateTime.parse(strDate, DateUtils.FORMATTER)
        ;

    List<EmployeeModel> employeeModels = employeeService.
        getEmployeesByDate(date);

    List<EmployeeResponse> employees = new ArrayList<>();
    for(EmployeeModel empployeeModel: employeeModels) {
            EmployeeResponse employee = employeeMapper.toResponse(
                empployeeModel);
            employees.add(employee);
    }

    return Response.ok(employees)
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
}

@POST()
@Path("on-duty/date")
public Response findEmployeesOnDuty(FindEmployeesByDateRequest request) {

    String strDate = request.getDate();
    LocalDateTime date = LocalDateTime.parse(strDate, DateUtils.FORMATTER)
        ;

    List<EmployeeModel> employeeModels = employeeService.
        getEmployeesByDateAndHolidayStatus(date, HolidayStatusEnum.ON_DUTY)
        ;

    List<EmployeeResponse> employees = new ArrayList<>();
    for(EmployeeModel empployeeModel: employeeModels) {
            EmployeeResponse employee = employeeMapper.toResponse(
                empployeeModel);
            employees.add(employee);
    }

    return Response.ok(employees)
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
```

```
    }

    @POST()
    @Path("on-holiday/date")
    public Response findEmployeesOnHoliday(FindEmployeesByDateRequest request)
        {

        String strDate = request.getDate();
        LocalDateTime date = LocalDateTime.parse(strDate, DateUtils.FORMATTER)
            ;

        List<EmployeeModel> employeeModels = employeeService.
            getEmployeesByDateAndHolidayStatus(date, HolidayStatusEnum.
            ON_HOLIDAY);

        List<EmployeeResponse> employees = new ArrayList<>();
        for(EmployeeModel empployeeModel: employeeModels) {
                EmployeeResponse employee = employeeMapper.toResponse(
                    empployeeModel);
                employees.add(employee);
        }

        return Response.ok(employees)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;
```

```
/**
 * REST API for checking the system is running
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/hello")
@RequestScoped
public class HelloResource {

        @GET
        @Produces(MediaType.APPLICATION_JSON)
        public Response get() {

                return Response.status(Response.Status.OK)
                                .entity("Vacation system is running")
                                .header("Access-Control-Allow-Origin", "*")
                                .build();
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.POST;
import javax.ws.rs.PUT;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.QueryParam;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;
import javax.ws.rs.core.Response.Status;

import org.jboss.logging.Logger;
```

```
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

import com.holidaysystem.common.DateUtils;
import com.holidaysystem.entity.HolidayRequestEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.mapper.HolidayRequestMapper;
import com.holidaysystem.model.HolidayRequestModel;
import com.holidaysystem.model.PrioritizedRequestModel;
import com.holidaysystem.vo.HolidayRequest;
import com.holidaysystem.vo.HolidayResponse;
import com.holidaysystem.vo.PrioritizedRecordsRequest;
import com.holidaysystem.service.IHolidayRequestService;

/**
 * REST API for holiday request resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/holiday-request")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
//@ServletSecurity(value = @HttpConstraint(rolesAllowed = {"admin_role"}))
public class HolidayRequestResource {

        private static final Logger logger = Logger.getLogger(
            HolidayRequestResource.class);

        @Inject
        private IHolidayRequestService holidayRequestService;
        @Inject
    private HolidayRequestMapper holidayRequestMapper;

        /**
         * Gets all holiday requests
         * limit 100
         * @return
         */
    @GET
    @Path("/all")
    public Response getHolidayRequests() {

        List<HolidayRequestModel> models = holidayRequestService.
            getHolidayRequests();

        List<HolidayResponse> holidayRequestResponses = new ArrayList<>();
        for(HolidayRequestModel model: models) {
                HolidayResponse holidayResponse = holidayRequestMapper.
                    toResponse(model);
                holidayRequestResponses.add(holidayResponse);
```

```
    }

    return Response.ok(holidayRequestResponses)
                    .build();
}

/**
 * Gets pages of holiday requests sorted by created by default
 * @param offset skip number of requests
 * @param limit size of the page
 * @return list of HolidayResponse
 */
@GET
@Path("/query")
public Response getHolidayRequestPages(@QueryParam("offset") int offset,
            @QueryParam("limit") int limit) {

    List<HolidayRequestModel> models = holidayRequestService.
        getHolidayRequests(offset, limit);

    List<HolidayResponse> holidayRequestResponses = new ArrayList<>();
    for(HolidayRequestModel model: models) {
            HolidayResponse holidayResponse = holidayRequestMapper.
                toResponse(model);
            holidayRequestResponses.add(holidayResponse);
    }

    return Response.ok(holidayRequestResponses)
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
}

@GET
@Path("/all/{status}")
public Response getHolidayRequests(@PathParam("status")
    HolidayRequestStatusEnum status) {

    logger.info("getHolidayRequests() with status " + status.name());

    List<HolidayRequestModel> models = holidayRequestService.
        getHolidayRequestsByStatus(status);

    List<HolidayResponse> holidayRequestResponses = new ArrayList<>();

    for(HolidayRequestModel model: models) {
            HolidayResponse holidayResponse = holidayRequestMapper.
                toResponse(model);
            holidayRequestResponses.add(holidayResponse);
    }

    return Response.ok(holidayRequestResponses)
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
}
```

```
@GET
@Path("/all/prioritized")
public Response getPrioritizedHolidayRequests(PrioritizedRecordsRequest
    request) {

    LocalDateTime date = LocalDateTime.parse(request.getDate(), DateUtils.
        FORMATTER);
    HolidayRequestStatusEnum requestStatus = HolidayRequestStatusEnum.
        valueOf(request.getRequestStatus());

    List<PrioritizedRequestModel> models = holidayRequestService
                    .getPrioritizedHolidayRequests(date, requestStatus);

    List<UUID> requestIds = new ArrayList<>();

    for(PrioritizedRequestModel model: models) {
            requestIds.add(model.getRequestId());
    }

    return Response.ok(requestIds)
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
}

@GET
@Path("/{id}/alternative-dates")
public Response getAlternativeDates(@PathParam("id") UUID id) {

    //holidayRequestService.getAlternativeDates(id);

    return Response.ok()
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
}

@GET()
@Path("/{id}")
public Response getHolidayRequestById(@PathParam("id") UUID id) {

    HolidayRequestModel model = holidayRequestService.fetchModelById(id);

    HolidayResponse holidayRequestResp = holidayRequestMapper.toResponse(
        model);

    return Response.ok(holidayRequestResp)
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
}

@POST()
@Consumes(MediaType.APPLICATION_JSON)
public Response createHolidayRequest(HolidayRequest holidayRequest) {
    UUID id = UUID.randomUUID();
```

```
        holidayRequestService.addHolidayRequest(id, holidayRequest);

        return Response.ok(id)
                        //.header("Access-Control-Allow-Origin", "*")
                        .status(Status.CREATED)
                        .build();
    }

    @PUT()
    @Path("/{id}")
    public Response updateHolidayRequest(@PathParam("id") UUID id,
        HolidayRequest holidayRequest) {
        HolidayRequestEntity updatedEntity = holidayRequestService.update(id,
            holidayRequest);

        if(updatedEntity != null) {
                HolidayResponse holidayRequestResp = holidayRequestMapper.
                    toResponse(updatedEntity);

                return Response.ok(holidayRequestResp)
                                .header("Access-Control-Allow-Origin", "*")
                                .build();
        }

        return Response.ok(null)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();

    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
```

```
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;

import org.jboss.logging.Logger;

import java.util.ArrayList;
import java.util.List;

import com.holidaysystem.enumeration.EmployeeRoleEnum;

/**
 * REST API for roles resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/role")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
//@ServletSecurity(value = @HttpConstraint(rolesAllowed = {"admin_role"}))
public class RolesResource {

        private static final Logger logger = Logger.getLogger(RolesResource.
            class);

    @GET
    @Path("/all")
    public Response getRoles() {

        logger.debug("getRoles()");

        List<String> roles = new ArrayList<>();

        for(EmployeeRoleEnum value: EmployeeRoleEnum.values()) {
                roles.add(value.name());
        }

        return Response.ok(roles)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
    }

}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
     eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
```

```
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;

import org.jboss.logging.Logger;

import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

import com.holidaysystem.entity.RequestAlertEntity;
import com.holidaysystem.mapper.RequestAlertMapper;
import com.holidaysystem.vo.RequestAlertResponse;
import com.holidaysystem.repository.RequestAlertRepository;

/**
 * REST API for request-alert resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/request-alert")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
//@ServletSecurity(value = @HttpConstraint(rolesAllowed = {"admin_role"}))
public class RequestAlertResource {

        private static final Logger logger = Logger.getLogger(
            RequestAlertResource.class);

    @Inject
    private RequestAlertRepository requestAlertRepository;
    @Inject
    private RequestAlertMapper requestAlertMapper;

    @GET
    @Path("/all")
```

```
    public Response getRequestsAlerts() {
        logger.debug("getRequestsAlerts()");

        List<RequestAlertEntity> entities = requestAlertRepository.
            getRequestAlerts();
        List<RequestAlertResponse> requestAlertResponses = new ArrayList<>();
        for(RequestAlertEntity entity: entities) {
                RequestAlertResponse holidayResponse = requestAlertMapper.
                    toResponse(entity);
                requestAlertResponses.add(holidayResponse);
        }

        return Response.ok(requestAlertResponses)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
    }

    @GET()
    @Path("/{id}")
    public Response getHolidayRequestById(@PathParam("id") UUID id) {
        RequestAlertEntity entity = requestAlertRepository.findById(id);
        RequestAlertResponse requestAlertResp = requestAlertMapper.toResponse(
            entity);
        return Response.ok(requestAlertResp)
                        .header("Access-Control-Allow-Origin", "*")
                        .build();
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.resource;

import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
```

```
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;

import org.jboss.logging.Logger;

import java.util.UUID;

import com.holidaysystem.vo.RequestValidationResponse;
import com.holidaysystem.service.IHolidayRequestService;

/**
 * REST API for holiday request resource
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Path("/request-processing")
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@RequestScoped
//@ServletSecurity(value = @HttpConstraint(rolesAllowed = {"admin_role"}))
public class RequestProcessingResource {

        private static final Logger logger = Logger.getLogger(
            RequestProcessingResource.class);

    @Inject
    private IHolidayRequestService holidayRequestService;

    @GET()
    @Path("/{id}/validate")
    public Response validate(@PathParam("id") UUID holidayRequestId) {

        logger.debug("validate request id " + holidayRequestId);

        boolean valid = holidayRequestService.validate(holidayRequestId);

        RequestValidationResponse resp = new RequestValidationResponse();
        resp.setRequestId(holidayRequestId);
        resp.setValid(valid);

        return Response.ok(resp)
                       .header("Access-Control-Allow-Origin", "*")
                       .build();
    }
}



/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
```

```
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.security;

import java.security.Principal;

import com.holidaysystem.model.AccountDetailsModel;

public class AppPrincipal implements Principal {

    private AccountDetailsModel accountDetail;

    public AppPrincipal(AccountDetailsModel accountDetail) {
        this.accountDetail = accountDetail;
    }

    @Override
    public String getName() {
        return accountDetail.getEmail();
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.security;

import java.util.HashSet;
import java.util.Set;
import javax.inject.Inject;
import javax.annotation.Resource;
```

```
import javax.security.enterprise.AuthenticationException;
import javax.security.enterprise.AuthenticationStatus;
import javax.security.enterprise.authentication.mechanism.http.
    HttpAuthenticationMechanism;
import javax.security.enterprise.authentication.mechanism.http.
    HttpMessageContext;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.sql.DataSource;
import javax.transaction.Transactional;

import com.holidaysystem.Constants;
import com.holidaysystem.entity.AccountEntity;
import com.holidaysystem.entity.AuthorizationRoleEntity;
import com.holidaysystem.model.AccountDetailsModel;
import com.holidaysystem.enumeration.AuthorizationRoleEnum;
import com.holidaysystem.repository.AccountRepository;
import com.holidaysystem.repository.AuthorizationRoleRepository;

/**
 * Authenticaiton mechanism using holidaysystem database
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
//@ApplicationScoped
public class AuthenticationMechanism implements HttpAuthenticationMechanism {
        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    private DataSource dataSource;

    @Override
    public AuthenticationStatus validateRequest(HttpServletRequest
        httpServletRequest,
                                                HttpServletResponse
                                                    httpServletResponse,
                                                HttpMessageContext
                                                    httpMessageContext) throws
                                                    AuthenticationException {

        String authHeader = httpServletRequest.getHeader("Authorization");

        if(authHeader.isBlank())
                return httpMessageContext.responseUnauthorized();

        String[] params = authHeader.split("&");

        if(params.length != 2)
                return httpMessageContext.responseUnauthorized();

        String username = params[0];
        String password = params[1];

        AccountDetailsModel accountDetails = findByEmailAndPassword(username,
            password);
```

```
        if (accountDetails != null) {

                Set<String> roles = new HashSet<>();
                roles.add(accountDetails.getAuthRole().name());

            return httpMessageContext.notifyContainerAboutLogin(
                    new AppPrincipal(accountDetails),
                    roles);
        }

        return httpMessageContext.responseUnauthorized();
    }

    //find why Inject does not work and fix deps
    @Transactional
    public AccountDetailsModel findByEmailAndPassword(String email, String
        password) {

        AccountRepository accountRepository = new AccountRepository();
        AuthorizationRoleRepository authRoleRepository = new
            AuthorizationRoleRepository();

                final String hashedPassword = accountRepository.
                    generateHashedPassword(password);

                AccountEntity account = accountRepository.
                    findByEmailAndPassword(email, hashedPassword);

        if(account == null)
                throw new RuntimeException("account not found");

        AuthorizationRoleEntity authRole = authRoleRepository.findById(account
            .getAuthRoleId());

        AccountDetailsModel accountDetails = new AccountDetailsModel();
        accountDetails.setId(account.getId());
        accountDetails.setEmail(account.getEmail());
        accountDetails.setActive(account.getActive());
        accountDetails.setAuthRole(AuthorizationRoleEnum.valueOf(authRole.
            getName()));

        return accountDetails;
        }
}
```

] package com.holidaysystem.security;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
```

```
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.security;

import java.security.Principal;

import com.holidaysystem.model.AccountDetailsModel;

public class AppPrincipal implements Principal {

    private AccountDetailsModel accountDetail;

    public AppPrincipal(AccountDetailsModel accountDetail) {
        this.accountDetail = accountDetail;
    }

    @Override
    public String getName() {
        return accountDetail.getEmail();
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.security;

import java.util.HashSet;
import java.util.Set;
import javax.inject.Inject;
import javax.annotation.Resource;
import javax.security.enterprise.AuthenticationException;
import javax.security.enterprise.AuthenticationStatus;
```

```
import javax.security.enterprise.authentication.mechanism.http.
    HttpAuthenticationMechanism;
import javax.security.enterprise.authentication.mechanism.http.
    HttpMessageContext;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.sql.DataSource;
import javax.transaction.Transactional;

import com.holidaysystem.Constants;
import com.holidaysystem.entity.AccountEntity;
import com.holidaysystem.entity.AuthorizationRoleEntity;
import com.holidaysystem.model.AccountDetailsModel;
import com.holidaysystem.enumeration.AuthorizationRoleEnum;
import com.holidaysystem.repository.AccountRepository;
import com.holidaysystem.repository.AuthorizationRoleRepository;

/**
 * Authenticaiton mechanism using holidaysystem database
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
//@ApplicationScoped
public class AuthenticationMechanism implements HttpAuthenticationMechanism {
        @Resource(lookup = Constants.DATASOURCE_LOOKUP_KEY)
    private DataSource dataSource;

    @Override
    public AuthenticationStatus validateRequest(HttpServletRequest
        httpServletRequest,
                                                HttpServletResponse
                                                    httpServletResponse,
                                                HttpMessageContext
                                                    httpMessageContext) throws
                                                    AuthenticationException {

        String authHeader = httpServletRequest.getHeader("Authorization");

        if(authHeader.isBlank())
                return httpMessageContext.responseUnauthorized();

        String[] params = authHeader.split("&");

        if(params.length != 2)
                return httpMessageContext.responseUnauthorized();

        String username = params[0];
        String password = params[1];

        AccountDetailsModel accountDetails = findByEmailAndPassword(username,
            password);

        if (accountDetails != null) {
```

```
                Set<String> roles = new HashSet<>();
                roles.add(accountDetails.getAuthRole().name());

            return httpMessageContext.notifyContainerAboutLogin(
                    new AppPrincipal(accountDetails),
                    roles);
        }

        return httpMessageContext.responseUnauthorized();
    }

    //find why Inject does not work and fix deps
    @Transactional
    public AccountDetailsModel findByEmailAndPassword(String email, String
        password) {

        AccountRepository accountRepository = new AccountRepository();
        AuthorizationRoleRepository authRoleRepository = new
            AuthorizationRoleRepository();

                final String hashedPassword = accountRepository.
                    generateHashedPassword(password);

                AccountEntity account = accountRepository.
                    findByEmailAndPassword(email, hashedPassword);

        if(account == null)
                throw new RuntimeException("account not found");

        AuthorizationRoleEntity authRole = authRoleRepository.findById(account
            .getAuthRoleId());

        AccountDetailsModel accountDetails = new AccountDetailsModel();
        accountDetails.setId(account.getId());
        accountDetails.setEmail(account.getEmail());
        accountDetails.setActive(account.getActive());
        accountDetails.setAuthRole(AuthorizationRoleEnum.valueOf(authRole.
            getName()));

        return accountDetails;
        }
}


/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
```

```
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.service;

import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.transaction.Transactional;

import com.holidaysystem.common.exception.RecordDuplicateException;
import com.holidaysystem.common.exception.RecordNotFoundException;
import com.holidaysystem.entity.AccountEntity;
import com.holidaysystem.entity.AuthorizationRoleEntity;
import com.holidaysystem.model.AccountDetailsModel;
import com.holidaysystem.enumeration.AuthorizationRoleEnum;
import com.holidaysystem.mapper.AccountMapper;
import com.holidaysystem.repository.IAccountRepository;
import com.holidaysystem.repository.IAuthorizationRoleRepository;
import com.holidaysystem.vo.RegistrationRequest;

import javax.inject.Inject;

/**
 * Auth service provides user register and login
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class AuthService implements IAuthService {

        @Inject
        IAccountRepository accountRepository;
        @Inject
        IAuthorizationRoleRepository authRoleRepository;
        @Inject
        AccountMapper accountMapper;

        @Transactional
        public AccountDetailsModel login(String email, String password) {

                AccountEntity account = accountRepository.
                    findByEmailAndPassword(email, password);

        if(account == null)
                throw new RecordNotFoundException("account not found");

        AuthorizationRoleEntity authRole = authRoleRepository.findById(account
```

```
            .getAuthRoleId());

        AccountDetailsModel accountDetails = accountMapper.toModel(account,
            authRole);

        return accountDetails;
        }

        @Transactional
        public String generateHash(String password) {
        return accountRepository.generateHashedPassword(password);
    }

        @Transactional
        public AccountDetailsModel register(RegistrationRequest
            registrationRequest) {

                AccountEntity dbEntity = accountRepository.findByEmail(
                    registrationRequest.getEmail());
                if(dbEntity != null)
                        throw new RecordDuplicateException("Email is used by
                            other account");

                final UUID id = UUID.randomUUID();
                final String hashedPassWithSalt = generateHash(
                    registrationRequest.getPassword());

                AccountEntity account = accountMapper.toEntity(id,
                    registrationRequest, hashedPassWithSalt);
                AuthorizationRoleEntity authRole = authRoleRepository.
                    findByName(AuthorizationRoleEnum.USER.name());
                account.setAuthRoleId(authRole.getId());

        accountRepository.save(account);

        AccountDetailsModel accountDetails = accountMapper.toModel(account,
            authRole);

        return accountDetails;
        }

}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
```

```
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.service;

import java.time.LocalDateTime;
import java.util.List;
import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.inject.Inject;
import javax.transaction.Transactional;

import com.holidaysystem.common.exception.RecordNotFoundException;
import com.holidaysystem.entity.AccountEntity;
import com.holidaysystem.entity.EmployeeEntity;
import com.holidaysystem.entity.HolidayDetailsEntity;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.mapper.EmployeeMapper;
import com.holidaysystem.model.EmployeeModel;
import com.holidaysystem.repository.IAccountRepository;
import com.holidaysystem.repository.IEmployeeRepository;
import com.holidaysystem.repository.IHolidayDetailsRepository;
import com.holidaysystem.vo.NewEmployeeRequest;

/**
 * Employee service provides employee details
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class EmployeeService implements IEmployeeService {
        @Inject
        IEmployeeRepository employeeRepository;
        @Inject
        IAccountRepository accountRepository;
        @Inject
        IHolidayDetailsRepository holidayDetailsRepository;
        @Inject
        EmployeeMapper employeeMapper;

    @Transactional
    public List<EmployeeModel> getEmployees() {

        List<EmployeeModel> employeeModels = employeeRepository.
            getEmployeeModels();

        return employeeModels;
    }
```

```
@Transactional
public List<EmployeeModel> getEmployeesByDate(LocalDateTime date) {

    List<EmployeeModel> employeeModels = employeeRepository.
        getEmployeeModelsOnHolidaysByDate(date);

    return employeeModels;
}

@Transactional
public List<EmployeeModel> getEmployeesByDateAndHolidayStatus(
    LocalDateTime date, HolidayStatusEnum holidayStatus) {

    List<EmployeeModel> employeeModels = employeeRepository.
        getEmployeeModelsByDateAndHolidayStatus(date, holidayStatus);

    return employeeModels;
}

@Transactional
public EmployeeModel findById(UUID employeeId) {

    EmployeeEntity employeeEntity = employeeRepository.findById(employeeId
        );

    if(employeeEntity == null)
            throw new RecordNotFoundException("Employee not found");

    AccountEntity accountEntity = accountRepository.findById(
        employeeEntity.getAccountId());
    HolidayDetailsEntity holidayDetailsEntity = holidayDetailsRepository.
        findByEmployeeId(employeeId);

    EmployeeModel empployeeModel = employeeMapper.toEmployeeModel(
        employeeEntity, accountEntity, holidayDetailsEntity);

    return empployeeModel;
}

@Transactional
public EmployeeModel create(NewEmployeeRequest employeeRequest) {

    EmployeeEntity employeeEntity = employeeMapper.toEntity(
        employeeRequest);
    boolean saved = employeeRepository.save(employeeEntity);

    if(!saved)
            throw new RuntimeException("Failed to add an employee");

    HolidayDetailsEntity holidayDetails = new HolidayDetailsEntity();
    holidayDetails.setId(UUID.randomUUID());
    holidayDetails.setEmployeeId(employeeEntity.getId());
    holidayDetails.setTotalDays(30);
```

```
        holidayDetails.setTakenDays(0);
        holidayDetails.setStatus(HolidayStatusEnum.ON_DUTY.name());
        holidayDetails.setCreated(LocalDateTime.now());
        holidayDetails.setModified(LocalDateTime.now());

        holidayDetailsRepository.save(holidayDetails);

        AccountEntity accountEntity = accountRepository.findById(
            employeeEntity.getAccountId());

        if(accountEntity == null)
                throw new RecordNotFoundException("Account not found");

        EmployeeModel empployeeModel = employeeMapper.toEmployeeModel(
            employeeEntity, accountEntity, holidayDetails);

        return empployeeModel;
    }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.service;

import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.LocalTime;
import java.time.temporal.ChronoUnit;
import java.util.Collections;
import java.util.LinkedList;
import java.util.List;
import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.transaction.Transactional;

import org.jboss.logging.Logger;
```

```
import com.holidaysystem.Constants;
import com.holidaysystem.common.exception.RecordNotFoundException;
import com.holidaysystem.comparator.HolidayRequestModelComparator;
import com.holidaysystem.constraints.IConstraintsValidator;
import com.holidaysystem.entity.EmployeeEntity;
import com.holidaysystem.entity.HolidayRequestEntity;
import com.holidaysystem.model.AlternativeDatesModel;
import com.holidaysystem.model.EmployeeModel;
import com.holidaysystem.model.HolidayRequestModel;
import com.holidaysystem.model.PrioritizedRequestModel;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.mapper.HolidayRequestMapper;
import com.holidaysystem.message.HolidayRequestMQProducer;
import com.holidaysystem.repository.IEmployeeRepository;
import com.holidaysystem.repository.IHolidayDetailsRepository;
import com.holidaysystem.repository.IHolidayRequestRepository;
import com.holidaysystem.vo.HolidayRequest;

import javax.inject.Inject;

/**
 * HolidayRequest service provides user register and login
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class HolidayRequestService implements IHolidayRequestService {

        private static final Logger logger = Logger.getLogger(
           HolidayRequestService.class);

        @Inject
    private IHolidayRequestRepository holidayRequestRepository;
        @Inject
    private IHolidayDetailsRepository holidayDetailsRepository;
        @Inject
    private IEmployeeRepository employeeRepository;
    @Inject
    private HolidayRequestMapper holidayRequestMapper;
    @Inject
    private HolidayRequestMQProducer holidayRequestMQProducer;
    @Inject
    private IConstraintsValidator constraintValidator;
    @Inject
    private HolidayRequestModelComparator comparator;

    @Transactional
        public List<HolidayRequestModel> getHolidayRequests() {
                List<HolidayRequestModel> models = holidayRequestRepository.
                   getHolidayRequestModels();
        return models;
        }
```

```
@Transactional
public List<HolidayRequestModel> getHolidayRequests(int offset, int
    limit) {
        List<HolidayRequestModel> models = holidayRequestRepository.
            getHolidayRequestModels(offset, limit);
return models;
}

@Transactional
public List<HolidayRequestModel> getHolidayRequestsByStatus(
    HolidayRequestStatusEnum status) {
        List<HolidayRequestModel> models = holidayRequestRepository.
            getHolidayRequestModelsByStatus(status);
        return models;
}

@Transactional
public HolidayRequestEntity findEntityById(UUID requestId) {
        HolidayRequestEntity entity = holidayRequestRepository.
            findById(requestId);
return entity;
}

@Transactional
public HolidayRequestModel fetchModelById(UUID requestId) {
        HolidayRequestEntity entity = holidayRequestRepository.
            findById(requestId);
        HolidayRequestModel model = holidayRequestMapper.toModel(
            entity);
return model;
}

@Transactional
public void addHolidayRequest(UUID requestId, HolidayRequest
    holidayRequest) {
        HolidayRequestEntity entity = holidayRequestMapper.toEntity(
            requestId, holidayRequest);
holidayRequestRepository.save(entity);
holidayRequestMQProducer.publish(requestId, holidayRequest);
}

@Transactional
public HolidayRequestEntity update(UUID requestId, HolidayRequest
    holidayRequest) {
        HolidayRequestEntity dbEntity = this.findEntityById(requestId)
            ;

if(dbEntity != null) {
        HolidayRequestEntity entity = holidayRequestMapper.toEntity(
            holidayRequest, dbEntity);
        holidayRequestRepository.update(requestId, entity);

        return entity;
}
```

```
logger.info("request with id: " + requestId + " not found");
throw new RecordNotFoundException("Holiday request is not found");
}

@Transactional
public boolean validate(UUID requestId) {

        HolidayRequestEntity dbHolidayRequestEntity = this.
            findEntityById(requestId);
        if(dbHolidayRequestEntity == null) {
                logger.error("holiday request not found");
                throw new RecordNotFoundException("holiday request not
                     found");
        }

        EmployeeEntity dbEmployeeEntity = employeeRepository.findById(
            dbHolidayRequestEntity.getEmployeeId());
        if(dbEmployeeEntity == null) {
                logger.error("employee not found");
                throw new RecordNotFoundException("employee not found
                    ");
        }

        List<EmployeeModel> employeeModels = employeeRepository.
            getEmployeeModelsByDepartmentId(dbEmployeeEntity.
            getDepartment());
        LocalDateTime now = LocalDateTime.now();
        int year = LocalDate.now().getYear();
        LocalDate dec22 = LocalDate.of(year, 12, 22);
        LocalDate jan3 = LocalDate.of(year + 1, 1, 3);

        if(now.isAfter(LocalDateTime.of(dec22, LocalTime.of(0, 0))) &&
                        now.isBefore(LocalDateTime.of(jan3, LocalTime.
                            of(0, 0)))) {

                final int percentOnDuty = LocalDate.now().
                    getMonthValue() == 8 ?
                                Constants.
                                    PERCENT_MEMBERS_AUGUST_ON_DUTY:
                        Constants.PERCENT_MEMBERS_ON_DUTY;

        return constraintValidator.hasHeadOrDeputyHead(employeeModels)
            &&
                        constraintValidator.hasManagerAndSeniorMemeber
                            (employeeModels) &&
                        constraintValidator.
                            hasSpecificPercentMemebersOnDuty(
                            employeeModels, percentOnDuty);
        }

        return true;
}
```

```
        @Override
        public List<PrioritizedRequestModel> getPrioritizedHolidayRequests(
            LocalDateTime date, HolidayRequestStatusEnum requestStatus) {
                List<PrioritizedRequestModel> prioritizedRequests = new
                    LinkedList<>();

                List<HolidayRequestModel> holidayRequests =
                    holidayDetailsRepository
                                .getApprovedAndByDate(date, requestStatus);

                holidayRequests.forEach(model -> {
                        int days = (int)model.getStartDate().until(model.
                            getEndDate(), ChronoUnit.DAYS);
                        model.setRequestedDays(days);
                });

                Collections.sort(holidayRequests, comparator);
                holidayRequests.forEach(holidayRequest -> {
                        PrioritizedRequestModel model = new
                            PrioritizedRequestModel();
                        model.setRequestId(holidayRequest.getId());
                        model.setRequestedDays(model.getRequestedDays());
                        model.setTakenDays(model.getTakenDays());

                        prioritizedRequests.add(model);
                });

                return prioritizedRequests;
        }

        @Override
        public List<AlternativeDatesModel> getAlternativeDates(UUID requestId)
            {

                return null;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
```

```
 */
package com.holidaysystem.service;

import com.holidaysystem.model.AccountDetailsModel;
import com.holidaysystem.vo.RegistrationRequest;

/**
 * Interface for Auth service provides user register and login
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IAuthService {
        /**
         * Login
         * @param email user email
         * @param password user password
         * @return AccountDetailsModel model
         */
        AccountDetailsModel login(String email, String password);
        /**
         * Generates hash of the password
         * @param password user password
         * @return hash of the password
         */
        String generateHash(String password);
        /**
         * Registers the new account
         * @param registrationRequest RegistrationRequest request model
         * @return AccountDetailsModel model
         */
        AccountDetailsModel register(RegistrationRequest registrationRequest);

}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.service;

import java.time.LocalDateTime;
```

```
import java.util.List;
import java.util.UUID;

import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.model.EmployeeModel;
import com.holidaysystem.vo.NewEmployeeRequest;

/**
 * Interface for Employee service provides employee details
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IEmployeeService {
        /**
         * Gets employees from all departments
         * @return list of EmployeeModel models
         */
    List<EmployeeModel> getEmployees();
    /**
     * Gets employees from all departments
     * @param date
     * @return
     */
    List<EmployeeModel> getEmployeesByDate(LocalDateTime date);
    /**
     *
     * @param date
     * @param holidayStatus
     * @return
     */
    List<EmployeeModel> getEmployeesByDateAndHolidayStatus(LocalDateTime date,
        HolidayStatusEnum holidayStatus);
    /**
     *
     * @param employeeId
     * @return
     */
    EmployeeModel findById(UUID employeeId);
    /**
     *
     * @param employeeRequest
     * @return
     */
    EmployeeModel create(NewEmployeeRequest employeeRequest);
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
```

```
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.service;

import java.time.LocalDateTime;
import java.util.List;
import java.util.UUID;

import com.holidaysystem.entity.HolidayRequestEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.model.AlternativeDatesModel;
import com.holidaysystem.model.HolidayRequestModel;
import com.holidaysystem.model.PrioritizedRequestModel;
import com.holidaysystem.vo.HolidayRequest;

/**
 * Interface for HolidayRequest service provides user register and login
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 */
public interface IHolidayRequestService {
        /**
         * Gets a list of holiday request models
         * limit 100
         * @return list of HolidayRequestModel
         */
        List<HolidayRequestModel> getHolidayRequests();
        /**
         * Gets a list of holiday request models sorted by created
         * @param offset skip records
         * @param limit size of the page
         * @return list of HolidayRequestModel
         */
        List<HolidayRequestModel> getHolidayRequests(int offset, int limit);
        /**
         * Gets a list of holiday request models by request status
         * @param status HolidayRequestStatusEnum status
         * @return list of HolidayRequestModel
         */
        List<HolidayRequestModel> getHolidayRequestsByStatus(
           HolidayRequestStatusEnum status);
        /**
         * Finds holiday request model, which extended data, by request id
         * @param requestId UUID request Id
         * @return holiday request model
         */
        HolidayRequestModel fetchModelById(UUID requestId);
        /**
```

```
         * Finds entity by request Id
         * @param requestId UUID request id
         * @return holiday request entity
         */
        HolidayRequestEntity findEntityById(UUID requestId);
        /**
         * Adds a new holiday request
         * @param requestId UUID holiday request
         * @param holidayRequest the model request
         */
        void addHolidayRequest(UUID requestId, HolidayRequest holidayRequest);
        /**
         *
         * @param requestId
         * @param holidayRequest
         * @return
         */
        HolidayRequestEntity update(UUID requestId, HolidayRequest
            holidayRequest);
        /**
         * Validates the request for supporting constraints
         * @param requestId UUID of the request
         * @return boolean: true is valid, false is not valid
         */
        boolean validate(UUID requestId);
        /**
         * Sorts requests by taken days and requested days
         * @param date
         * @param requestStatus
         * @return list of prioritized requests
         */
        List<PrioritizedRequestModel> getPrioritizedHolidayRequests(
            LocalDateTime date, HolidayRequestStatusEnum requestStatus);
        /**
         * Calculates alternative dates for broken requests
         * @param requestId UUID of the broken request
         * @return list of alternative dates
         */
        List<AlternativeDatesModel> getAlternativeDates(UUID requestId);
}
```

] package com.holidaysystem.service;

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
```

```
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.service;

import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.transaction.Transactional;

import com.holidaysystem.common.exception.RecordDuplicateException;
import com.holidaysystem.common.exception.RecordNotFoundException;
import com.holidaysystem.entity.AccountEntity;
import com.holidaysystem.entity.AuthorizationRoleEntity;
import com.holidaysystem.model.AccountDetailsModel;
import com.holidaysystem.enumeration.AuthorizationRoleEnum;
import com.holidaysystem.mapper.AccountMapper;
import com.holidaysystem.repository.IAccountRepository;
import com.holidaysystem.repository.IAuthorizationRoleRepository;
import com.holidaysystem.vo.RegistrationRequest;

import javax.inject.Inject;

/**
 * Auth service provides user register and login
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class AuthService implements IAuthService {

        @Inject
        IAccountRepository accountRepository;
        @Inject
        IAuthorizationRoleRepository authRoleRepository;
        @Inject
        AccountMapper accountMapper;

        @Transactional
        public AccountDetailsModel login(String email, String password) {

                AccountEntity account = accountRepository.
                    findByEmailAndPassword(email, password);

        if(account == null)
                throw new RecordNotFoundException("account not found");

        AuthorizationRoleEntity authRole = authRoleRepository.findById(account
            .getAuthRoleId());
```

```
        AccountDetailsModel accountDetails = accountMapper.toModel(account,
            authRole);

        return accountDetails;
        }

        @Transactional
        public String generateHash(String password) {
        return accountRepository.generateHashedPassword(password);
    }

        @Transactional
        public AccountDetailsModel register(RegistrationRequest
            registrationRequest) {

                AccountEntity dbEntity = accountRepository.findByEmail(
                    registrationRequest.getEmail());
                if(dbEntity != null)
                        throw new RecordDuplicateException("Email is used by
                            other account");

                final UUID id = UUID.randomUUID();
                final String hashedPassWithSalt = generateHash(
                    registrationRequest.getPassword());

                AccountEntity account = accountMapper.toEntity(id,
                    registrationRequest, hashedPassWithSalt);
                AuthorizationRoleEntity authRole = authRoleRepository.
                    findByName(AuthorizationRoleEnum.USER.name());
                account.setAuthRoleId(authRole.getId());

        accountRepository.save(account);

        AccountDetailsModel accountDetails = accountMapper.toModel(account,
            authRole);

        return accountDetails;
        }

}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
```

```
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.service;

import java.time.LocalDateTime;
import java.util.List;
import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.inject.Inject;
import javax.transaction.Transactional;

import com.holidaysystem.common.exception.RecordNotFoundException;
import com.holidaysystem.entity.AccountEntity;
import com.holidaysystem.entity.EmployeeEntity;
import com.holidaysystem.entity.HolidayDetailsEntity;
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.mapper.EmployeeMapper;
import com.holidaysystem.model.EmployeeModel;
import com.holidaysystem.repository.IAccountRepository;
import com.holidaysystem.repository.IEmployeeRepository;
import com.holidaysystem.repository.IHolidayDetailsRepository;
import com.holidaysystem.vo.NewEmployeeRequest;

/**
 * Employee service provides employee details
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class EmployeeService implements IEmployeeService {
        @Inject
        IEmployeeRepository employeeRepository;
        @Inject
        IAccountRepository accountRepository;
        @Inject
        IHolidayDetailsRepository holidayDetailsRepository;
        @Inject
        EmployeeMapper employeeMapper;

    @Transactional
    public List<EmployeeModel> getEmployees() {

        List<EmployeeModel> employeeModels = employeeRepository.
            getEmployeeModels();

        return employeeModels;
    }

    @Transactional
```

```
public List<EmployeeModel> getEmployeesByDate(LocalDateTime date) {

    List<EmployeeModel> employeeModels = employeeRepository.
        getEmployeeModelsOnHolidaysByDate(date);

    return employeeModels;
}

@Transactional
public List<EmployeeModel> getEmployeesByDateAndHolidayStatus(
    LocalDateTime date, HolidayStatusEnum holidayStatus) {

    List<EmployeeModel> employeeModels = employeeRepository.
        getEmployeeModelsByDateAndHolidayStatus(date, holidayStatus);

    return employeeModels;
}

@Transactional
public EmployeeModel findById(UUID employeeId) {

    EmployeeEntity employeeEntity = employeeRepository.findById(employeeId
        );

    if(employeeEntity == null)
            throw new RecordNotFoundException("Employee not found");

    AccountEntity accountEntity = accountRepository.findById(
        employeeEntity.getAccountId());
    HolidayDetailsEntity holidayDetailsEntity = holidayDetailsRepository.
        findByEmployeeId(employeeId);

    EmployeeModel empployeeModel = employeeMapper.toEmployeeModel(
        employeeEntity, accountEntity, holidayDetailsEntity);

    return empployeeModel;
}

@Transactional
public EmployeeModel create(NewEmployeeRequest employeeRequest) {

    EmployeeEntity employeeEntity = employeeMapper.toEntity(
        employeeRequest);
    boolean saved = employeeRepository.save(employeeEntity);

    if(!saved)
            throw new RuntimeException("Failed to add an employee");

    HolidayDetailsEntity holidayDetails = new HolidayDetailsEntity();
    holidayDetails.setId(UUID.randomUUID());
    holidayDetails.setEmployeeId(employeeEntity.getId());
    holidayDetails.setTotalDays(30);
    holidayDetails.setTakenDays(0);
    holidayDetails.setStatus(HolidayStatusEnum.ON_DUTY.name());
```

```
        holidayDetails.setCreated(LocalDateTime.now());
        holidayDetails.setModified(LocalDateTime.now());

        holidayDetailsRepository.save(holidayDetails);

        AccountEntity accountEntity = accountRepository.findById(
            employeeEntity.getAccountId());

        if(accountEntity == null)
                throw new RecordNotFoundException("Account not found");

        EmployeeModel empployeeModel = employeeMapper.toEmployeeModel(
            employeeEntity, accountEntity, holidayDetails);

        return empployeeModel;
    }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.service;

import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.LocalTime;
import java.time.temporal.ChronoUnit;
import java.util.Collections;
import java.util.LinkedList;
import java.util.List;
import java.util.UUID;

import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.inject.Default;
import javax.transaction.Transactional;

import org.jboss.logging.Logger;

import com.holidaysystem.Constants;
import com.holidaysystem.common.exception.RecordNotFoundException;
```

```
import com.holidaysystem.comparator.HolidayRequestModelComparator;
import com.holidaysystem.constraints.IConstraintsValidator;
import com.holidaysystem.entity.EmployeeEntity;
import com.holidaysystem.entity.HolidayRequestEntity;
import com.holidaysystem.model.AlternativeDatesModel;
import com.holidaysystem.model.EmployeeModel;
import com.holidaysystem.model.HolidayRequestModel;
import com.holidaysystem.model.PrioritizedRequestModel;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.mapper.HolidayRequestMapper;
import com.holidaysystem.message.HolidayRequestMQProducer;
import com.holidaysystem.repository.IEmployeeRepository;
import com.holidaysystem.repository.IHolidayDetailsRepository;
import com.holidaysystem.repository.IHolidayRequestRepository;
import com.holidaysystem.vo.HolidayRequest;

import javax.inject.Inject;

/**
 * HolidayRequest service provides user register and login
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@ApplicationScoped
@Default
public class HolidayRequestService implements IHolidayRequestService {

        private static final Logger logger = Logger.getLogger(
            HolidayRequestService.class);

        @Inject
    private IHolidayRequestRepository holidayRequestRepository;
        @Inject
    private IHolidayDetailsRepository holidayDetailsRepository;
        @Inject
    private IEmployeeRepository employeeRepository;
    @Inject
    private HolidayRequestMapper holidayRequestMapper;
    @Inject
    private HolidayRequestMQProducer holidayRequestMQProducer;
    @Inject
    private IConstraintsValidator constraintValidator;
    @Inject
    private HolidayRequestModelComparator comparator;

    @Transactional
        public List<HolidayRequestModel> getHolidayRequests() {
                List<HolidayRequestModel> models = holidayRequestRepository.
                    getHolidayRequestModels();
        return models;
        }

        @Transactional
        public List<HolidayRequestModel> getHolidayRequests(int offset, int
```

```
    limit) {
        List<HolidayRequestModel> models = holidayRequestRepository.
            getHolidayRequestModels(offset, limit);
return models;
}

@Transactional
public List<HolidayRequestModel> getHolidayRequestsByStatus(
    HolidayRequestStatusEnum status) {
        List<HolidayRequestModel> models = holidayRequestRepository.
            getHolidayRequestModelsByStatus(status);
        return models;
}

@Transactional
public HolidayRequestEntity findEntityById(UUID requestId) {
        HolidayRequestEntity entity = holidayRequestRepository.
            findById(requestId);
return entity;
}

@Transactional
public HolidayRequestModel fetchModelById(UUID requestId) {
        HolidayRequestEntity entity = holidayRequestRepository.
            findById(requestId);
        HolidayRequestModel model = holidayRequestMapper.toModel(
            entity);
return model;
}

@Transactional
public void addHolidayRequest(UUID requestId, HolidayRequest
    holidayRequest) {
        HolidayRequestEntity entity = holidayRequestMapper.toEntity(
            requestId, holidayRequest);
holidayRequestRepository.save(entity);
holidayRequestMQProducer.publish(requestId, holidayRequest);
}

@Transactional
public HolidayRequestEntity update(UUID requestId, HolidayRequest
    holidayRequest) {
        HolidayRequestEntity dbEntity = this.findEntityById(requestId)
            ;

if(dbEntity != null) {
        HolidayRequestEntity entity = holidayRequestMapper.toEntity(
            holidayRequest, dbEntity);
        holidayRequestRepository.update(requestId, entity);

        return entity;
}

logger.info("request with id: " + requestId + " not found");
```

```
throw new RecordNotFoundException("Holiday request is not found");
}

@Transactional
public boolean validate(UUID requestId) {

        HolidayRequestEntity dbHolidayRequestEntity = this.
            findEntityById(requestId);
        if(dbHolidayRequestEntity == null) {
                logger.error("holiday request not found");
                throw new RecordNotFoundException("holiday request not
                     found");
        }

        EmployeeEntity dbEmployeeEntity = employeeRepository.findById(
            dbHolidayRequestEntity.getEmployeeId());
        if(dbEmployeeEntity == null) {
                logger.error("employee not found");
                throw new RecordNotFoundException("employee not found
                    ");
        }

        List<EmployeeModel> employeeModels = employeeRepository.
            getEmployeeModelsByDepartmentId(dbEmployeeEntity.
            getDepartment());
        LocalDateTime now = LocalDateTime.now();
        int year = LocalDate.now().getYear();
        LocalDate dec22 = LocalDate.of(year, 12, 22);
        LocalDate jan3 = LocalDate.of(year + 1, 1, 3);

        if(now.isAfter(LocalDateTime.of(dec22, LocalTime.of(0, 0))) &&
                        now.isBefore(LocalDateTime.of(jan3, LocalTime.
                            of(0, 0)))) {

                final int percentOnDuty = LocalDate.now().
                    getMonthValue() == 8 ?
                                Constants.
                                    PERCENT_MEMBERS_AUGUST_ON_DUTY:
                        Constants.PERCENT_MEMBERS_ON_DUTY;

        return constraintValidator.hasHeadOrDeputyHead(employeeModels)
            &&
                        constraintValidator.hasManagerAndSeniorMemeber
                            (employeeModels) &&
                        constraintValidator.
                            hasSpecificPercentMemebersOnDuty(
                            employeeModels, percentOnDuty);
        }

        return true;
}

@Override
public List<PrioritizedRequestModel> getPrioritizedHolidayRequests(
```

```
            LocalDateTime date, HolidayRequestStatusEnum requestStatus) {
                List<PrioritizedRequestModel> prioritizedRequests = new
                    LinkedList<>();

                List<HolidayRequestModel> holidayRequests =
                    holidayDetailsRepository
                                .getApprovedAndByDate(date, requestStatus);

                holidayRequests.forEach(model -> {
                        int days = (int)model.getStartDate().until(model.
                            getEndDate(), ChronoUnit.DAYS);
                        model.setRequestedDays(days);
                });

                Collections.sort(holidayRequests, comparator);
                holidayRequests.forEach(holidayRequest -> {
                        PrioritizedRequestModel model = new
                            PrioritizedRequestModel();
                        model.setRequestId(holidayRequest.getId());
                        model.setRequestedDays(model.getRequestedDays());
                        model.setTakenDays(model.getTakenDays());

                        prioritizedRequests.add(model);
                });

                return prioritizedRequests;
        }

        @Override
        public List<AlternativeDatesModel> getAlternativeDates(UUID requestId)
            {

                return null;
        }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.service;
```

```
import com.holidaysystem.model.AccountDetailsModel;
import com.holidaysystem.vo.RegistrationRequest;

/**
 * Interface for Auth service provides user register and login
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IAuthService {
        /**
         * Login
         * @param email user email
         * @param password user password
         * @return AccountDetailsModel model
         */
        AccountDetailsModel login(String email, String password);
        /**
         * Generates hash of the password
         * @param password user password
         * @return hash of the password
         */
        String generateHash(String password);
        /**
         * Registers the new account
         * @param registrationRequest RegistrationRequest request model
         * @return AccountDetailsModel model
         */
        AccountDetailsModel register(RegistrationRequest registrationRequest);

}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.service;

import java.time.LocalDateTime;
import java.util.List;
import java.util.UUID;
```

```
import com.holidaysystem.enumeration.HolidayStatusEnum;
import com.holidaysystem.model.EmployeeModel;
import com.holidaysystem.vo.NewEmployeeRequest;

/**
 * Interface for Employee service provides employee details
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
public interface IEmployeeService {
        /**
         * Gets employees from all departments
         * @return list of EmployeeModel models
         */
    List<EmployeeModel> getEmployees();
    /**
     * Gets employees from all departments
     * @param date
     * @return
     */
    List<EmployeeModel> getEmployeesByDate(LocalDateTime date);
    /**
     *
     * @param date
     * @param holidayStatus
     * @return
     */
    List<EmployeeModel> getEmployeesByDateAndHolidayStatus(LocalDateTime date,
        HolidayStatusEnum holidayStatus);
    /**
     *
     * @param employeeId
     * @return
     */
    EmployeeModel findById(UUID employeeId);
    /**
     *
     * @param employeeRequest
     * @return
     */
    EmployeeModel create(NewEmployeeRequest employeeRequest);
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
```

```
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.service;

import java.time.LocalDateTime;
import java.util.List;
import java.util.UUID;

import com.holidaysystem.entity.HolidayRequestEntity;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
import com.holidaysystem.model.AlternativeDatesModel;
import com.holidaysystem.model.HolidayRequestModel;
import com.holidaysystem.model.PrioritizedRequestModel;
import com.holidaysystem.vo.HolidayRequest;

/**
 * Interface for HolidayRequest service provides user register and login
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 */
public interface IHolidayRequestService {
        /**
         * Gets a list of holiday request models
         * limit 100
         * @return list of HolidayRequestModel
         */
        List<HolidayRequestModel> getHolidayRequests();
        /**
         * Gets a list of holiday request models sorted by created
         * @param offset skip records
         * @param limit size of the page
         * @return list of HolidayRequestModel
         */
        List<HolidayRequestModel> getHolidayRequests(int offset, int limit);
        /**
         * Gets a list of holiday request models by request status
         * @param status HolidayRequestStatusEnum status
         * @return list of HolidayRequestModel
         */
        List<HolidayRequestModel> getHolidayRequestsByStatus(
           HolidayRequestStatusEnum status);
        /**
         * Finds holiday request model, which extended data, by request id
         * @param requestId UUID request Id
         * @return holiday request model
         */
        HolidayRequestModel fetchModelById(UUID requestId);
        /**
         * Finds entity by request Id
         * @param requestId UUID request id
```

```
         * @return holiday request entity
         */
        HolidayRequestEntity findEntityById(UUID requestId);
        /**
         * Adds a new holiday request
         * @param requestId UUID holiday request
         * @param holidayRequest the model request
         */
        void addHolidayRequest(UUID requestId, HolidayRequest holidayRequest);
        /**
         *
         * @param requestId
         * @param holidayRequest
         * @return
         */
        HolidayRequestEntity update(UUID requestId, HolidayRequest
            holidayRequest);
        /**
         * Validates the request for supporting constraints
         * @param requestId UUID of the request
         * @return boolean: true is valid, false is not valid
         */
        boolean validate(UUID requestId);
        /**
         * Sorts requests by taken days and requested days
         * @param date
         * @param requestStatus
         * @return list of prioritized requests
         */
        List<PrioritizedRequestModel> getPrioritizedHolidayRequests(
            LocalDateTime date, HolidayRequestStatusEnum requestStatus);
        /**
         * Calculates alternative dates for broken requests
         * @param requestId UUID of the broken request
         * @return list of alternative dates
         */
        List<AlternativeDatesModel> getAlternativeDates(UUID requestId);
}


/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
```

```
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.validation;

import static java.lang.annotation.ElementType.ANNOTATION_TYPE;
import static java.lang.annotation.ElementType.CONSTRUCTOR;
import static java.lang.annotation.ElementType.FIELD;
import static java.lang.annotation.ElementType.METHOD;
import static java.lang.annotation.ElementType.PARAMETER;
import static java.lang.annotation.ElementType.TYPE_USE;

import java.lang.annotation.Documented;
import java.lang.annotation.Inherited;
import java.lang.annotation.Retention;
import java.lang.annotation.Target;
import java.lang.annotation.RetentionPolicy;
import javax.validation.Payload;
import javax.enterprise.inject.Stereotype;
import javax.validation.Constraint;
import javax.ws.rs.ext.Provider;

import jakarta.inject.Qualifier;

/**
 * Annotation for validating that string has a valid enum value
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Qualifier
@Target({ METHOD, FIELD, ANNOTATION_TYPE, CONSTRUCTOR, PARAMETER, TYPE_USE })
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Constraint(validatedBy = ValueOfEnumValidator.class)
public @interface ValueOfEnum {
    Class<? extends Enum<?>> enumClass();
    String message() default "must be any of enum {enumClass}";
    Class<?>[] groups() default {};
    Class<? extends Payload>[] payload() default {};
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
```

```
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.validation;

import java.util.List;
import java.util.stream.Collectors;
import java.util.stream.Stream;

import javax.validation.ConstraintValidator;
import javax.validation.ConstraintValidatorContext;

public class ValueOfEnumValidator implements ConstraintValidator<ValueOfEnum,
    CharSequence> {
    private List<String> acceptedValues;

    @Override
    public void initialize(ValueOfEnum annotation) {
        acceptedValues = Stream.of(annotation.enumClass().getEnumConstants())
                .map(Enum::name)
                .collect(Collectors.toList());
    }

    @Override
    public boolean isValid(CharSequence value, ConstraintValidatorContext
        context) {
        if (value == null) {
            return true;
        }

        return acceptedValues.contains(value.toString());
    }
}
```

] package com.holidaysystem.validation;

```
/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.validation;
```

```
import static java.lang.annotation.ElementType.ANNOTATION_TYPE;
import static java.lang.annotation.ElementType.CONSTRUCTOR;
import static java.lang.annotation.ElementType.FIELD;
import static java.lang.annotation.ElementType.METHOD;
import static java.lang.annotation.ElementType.PARAMETER;
import static java.lang.annotation.ElementType.TYPE_USE;

import java.lang.annotation.Documented;
import java.lang.annotation.Inherited;
import java.lang.annotation.Retention;
import java.lang.annotation.Target;
import java.lang.annotation.RetentionPolicy;
import javax.validation.Payload;
import javax.enterprise.inject.Stereotype;
import javax.validation.Constraint;
import javax.ws.rs.ext.Provider;

import jakarta.inject.Qualifier;

/**
 * Annotation for validating that string has a valid enum value
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@Qualifier
@Target({ METHOD, FIELD, ANNOTATION_TYPE, CONSTRUCTOR, PARAMETER, TYPE_USE })
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Constraint(validatedBy = ValueOfEnumValidator.class)
public @interface ValueOfEnum {
    Class<? extends Enum<?>> enumClass();
    String message() default "must be any of enum {enumClass}";
    Class<?>[] groups() default {};
    Class<? extends Payload>[] payload() default {};
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
```

```
package com.holidaysystem.validation;

import java.util.List;
import java.util.stream.Collectors;
import java.util.stream.Stream;

import javax.validation.ConstraintValidator;
import javax.validation.ConstraintValidatorContext;

public class ValueOfEnumValidator implements ConstraintValidator<ValueOfEnum,
    CharSequence> {
    private List<String> acceptedValues;

    @Override
    public void initialize(ValueOfEnum annotation) {
        acceptedValues = Stream.of(annotation.enumClass().getEnumConstants())
                .map(Enum::name)
                .collect(Collectors.toList());
    }

    @Override
    public boolean isValid(CharSequence value, ConstraintValidatorContext
        context) {
        if (value == null) {
            return true;
        }

        return acceptedValues.contains(value.toString());
    }
}



/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.Email;
import javax.validation.constraints.NotBlank;
import javax.validation.constraints.PastOrPresent;
```

```
import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Response model for account
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class AccountResponse implements Serializable {

        @NotBlank
        private UUID id;

        @Email
        private String email;

        @NotBlank
    private String authRole;

        @NotBlank
    private Boolean active;

    public UUID getId() {
        return this.id;
    }
    public void setId(UUID id) {
        this.id = id;
    }
    public String getEmail() {
        return this.email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getAuthRole() {
        return this.authRole;
    }
    public void setAuthRole(String authRole) {
        this.authRole = authRole;
    }
    public Boolean getActive() {
        return this.active;
    }
    public void setActive(Boolean active) {
        this.active = active;
    }
}
```

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.PositiveOrZero;

import com.holidaysystem.enumeration.HolidayStatusEnum;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Response model for employee
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class EmployeeResponse implements Serializable {

        @NotBlank
        private UUID id;

        @NotBlank
        private UUID accountId;

    private String email;

    @NotBlank
    private String firstName;

    @NotBlank
    private String lastName;
```

```
@NotBlank
private String role;

@NotBlank
private String department;

@PositiveOrZero
private Integer years;

@PositiveOrZero
private Integer totalDays;

@PositiveOrZero
private Integer takenDays;

@NotBlank
private String holidayStatus;


public UUID getId() {
    return this.id;
}
public void setId(UUID id) {
    this.id = id;
}
public UUID getAccountId() {
    return this.accountId;
}
public void setAccountId(UUID accountId) {
    this.accountId = accountId;
}
public String getEmail() {
    return this.email;
}
public void setEmail(String email) {
    this.email = email;
}
public String getFirstName() {
    return this.firstName;
}
public void setFirstName(String firstName) {
    this.firstName = firstName;
}
public String getLastName() {
    return this.lastName;
}
public void setLastName(String lastName) {
    this.lastName = lastName;
}
public String getRole() {
    return this.role;
}
public void setRole(String role) {
```

```
        this.role = role;
    }
    public String getDepartment() {
        return this.department;
    }
    public void setDepartment(String department) {
        this.department = department;
    }
    public void setYears(Integer years) {this.years = years; }
    public Integer getYears() {return this.years;}
    public void setTotalDays(Integer totalDays) {this.totalDays = totalDays; }
    public Integer getTotalDays() {return this.totalDays;}
    public void setTakenDays(Integer takenDays) {this.takenDays = takenDays; }
    public Integer getTakenDays() {return this.takenDays;}
    public String getHolidayStatus() {
        return this.holidayStatus;
    }
    public void setHolidayStatus(String holidayStatus) {
        this.holidayStatus = holidayStatus;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.NotBlank;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;

/**
 * Request model for employees by date
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
```

```
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class FindEmployeesByDateRequest implements Serializable {
        private static final long serialVersionUID = 1L;

        @NotBlank
    private String date;

    public void setDate(String date) {
        this.date = date;
    }
    public String getDate() {return this.date; }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.NotBlank;

import com.holidaysystem.validation.ValueOfEnum;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Request model for holiday request
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class HolidayRequest implements Serializable {
```

```
    @NotBlank
    private UUID employeeId;

    @NotBlank
    @ValueOfEnum(enumClass = HolidayRequestStatusEnum.class)
    private String status;

    @NotBlank
    private String startDate;

    @NotBlank
    private String endDate;

    public UUID getEmployeeId() {
        return this.employeeId;
    }
    public void setEmployeeId(UUID employeeId) {
        this.employeeId = employeeId;
    }
    public String getStatus() {
        return this.status;
    }
    public void setStatus(String status) {
        this.status = status;
    }
    public String getStartDate() {
        return this.startDate;
    }
    public void setEndDate(String endDate) {
        this.endDate = endDate;
    }
    public String getEndDate() {
        return this.endDate;
    }
    public void setStartDate(String startDate) {
        this.startDate = startDate;
    }

}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
```

```
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.PositiveOrZero;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Response model for holiday request
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class HolidayResponse implements Serializable {

        @NotBlank
    private UUID id;

    @NotBlank
    private UUID employeeId;

    @NotBlank
    private String status;

    private Integer requestedDays;

    @NotBlank
    private String startDate;

    @NotBlank
    private String endDate;

    private Integer years;

    private Integer totalDays;

    private Integer takenDays;

    public UUID getId() {
        return this.id;
    }
    public void setId(UUID id) {
        this.id = id;
    }
```

```
    public UUID getEmployeeId() {
        return this.employeeId;
    }
    public void setEmployeeId(UUID employeeId) {
        this.employeeId = employeeId;
    }
    public String getStatus() {
        return this.status;
    }
    public void setStatus(String status) {
        this.status = status;
    }
    public String getStartDate() {
        return this.startDate;
    }
    public void setStartDate(String startDate) {
        this.startDate = startDate;
    }
    public void setEndDate(String endDate) {
        this.endDate = endDate;
    }
    public String getEndDate() {
        return this.endDate;
    }
    public void setYears(Integer years) {this.years = years; }
    public Integer getYears() {return this.years;}
    public void setTotalDays(Integer totalDays) {this.totalDays = totalDays; }
    public Integer getTotalDays() {return this.totalDays;}
    public void setTakenDays(Integer takenDays) {this.takenDays = takenDays; }
    public Integer getTakenDays() {return this.takenDays;}
    public void setRequestedDays(Integer requestedDays) {this.requestedDays =
       requestedDays; }
    public Integer getRequestedDays() {return this.requestedDays;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;
```

```
import javax.validation.constraints.Email;
import javax.validation.constraints.NotBlank;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;

/**
 * Request model for login
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class LoginRequest implements Serializable {

    @Email
    private String email;

    @NotBlank
    private String password;

    public String getEmail() {
        return this.email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getPassword() {
        return this.password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
```

```
 */
package com.holidaysystem.vo;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.PositiveOrZero;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Request model for employee
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class NewEmployeeRequest implements Serializable {

    @NotBlank
    private String firstName;

    @NotBlank
    private String lastName;

    @NotBlank
    private UUID accountId;

    @NotBlank
    private String role;

    @NotBlank
    private String department;

    @PositiveOrZero
    private Integer years;


    public String getFirstName() {
        return this.firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    public String getLastName() {
        return this.lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
    public UUID getAccountId() {
```

```
        return this.accountId;
    }
    public void setAccountId(UUID accountId) {
        this.accountId = accountId;
    }
    public String getRole() {
        return this.role;
    }
    public void setRole(String role) {
        this.role = role;
    }
    public String getDepartment() {
        return this.department;
    }
    public void setDepartment(String department) {
        this.department = department;
    }
    public void setYears(Integer years) {this.years = years; }
    public Integer getYears() {return this.years;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;

/**
 * Prioritized records request
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class PrioritizedRecordsRequest implements Serializable {
```

```
        private String date;

        private String requestStatus;

    public String getRequestStatus() {
        return this.requestStatus;
    }
    public void setRequestStatus(String requestStatus) {
        this.requestStatus = requestStatus;
    }
    public String getDate() {
        return this.date;
    }
    public void setDate(String date) {
        this.date = date;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.Email;
import javax.validation.constraints.NotBlank;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;

/**
 * Request model for registration
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
```

```
public class RegistrationRequest implements Serializable {

    @Email
    private String email;

    @NotBlank
    private String password;

    @NotBlank
    private String authRole;

    public String getEmail() {
        return this.email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getPassword() {
        return this.password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public String getAuthRole() {
        return this.authRole;
    }
    public void setAuthRole(String authRole) {
        this.authRole = authRole;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import java.io.Serializable;
import java.util.UUID;

import javax.validation.constraints.NotBlank;
```

```
import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

/**
 * Response model for request alert
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class RequestAlertResponse implements Serializable {

        @NotBlank
    private UUID id;

        @NotBlank
    private UUID requestId;

    @NotBlank
    private String date;

    public UUID getId() {
        return this.id;
    }
    public void setId(UUID id) {
        this.id = id;
    }
    public UUID getRequestId() {
        return this.requestId;
    }
    public void setRequestId(UUID requestId) {
        this.requestId = requestId;
    }
    public String getDate() {
        return this.date;
    }
    public void setDate(String date) {
        this.date = date;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
```

```
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import java.io.Serializable;
import java.util.UUID;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

/**
 * Response model for request alert
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class RequestValidationResponse implements Serializable {

        @NotBlank
    private UUID requestId;

        @NotNull
    private Boolean valid;

    public UUID getRequestId() {
        return this.requestId;
    }
    public void setRequestId(UUID requestId) {
        this.requestId = requestId;
    }
    public Boolean getValid() {
        return this.valid;
    }
    public void setValid(Boolean valid) {
        this.valid = valid;
    }
}
```

] package com.holidaysystem.vo;

```
/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
```

```
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.Email;
import javax.validation.constraints.NotBlank;
import javax.validation.constraints.PastOrPresent;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Response model for account
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class AccountResponse implements Serializable {

        @NotBlank
        private UUID id;

        @Email
        private String email;

        @NotBlank
    private String authRole;

        @NotBlank
    private Boolean active;

    public UUID getId() {
        return this.id;
    }
    public void setId(UUID id) {
        this.id = id;
    }
    public String getEmail() {
        return this.email;
    }
```

```
    public void setEmail(String email) {
        this.email = email;
    }
    public String getAuthRole() {
        return this.authRole;
    }
    public void setAuthRole(String authRole) {
        this.authRole = authRole;
    }
    public Boolean getActive() {
        return this.active;
    }
    public void setActive(Boolean active) {
        this.active = active;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.PositiveOrZero;

import com.holidaysystem.enumeration.HolidayStatusEnum;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Response model for employee
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
```

```
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class EmployeeResponse implements Serializable {

        @NotBlank
        private UUID id;

        @NotBlank
        private UUID accountId;

    private String email;

    @NotBlank
    private String firstName;

    @NotBlank
    private String lastName;

    @NotBlank
    private String role;

    @NotBlank
    private String department;

    @PositiveOrZero
    private Integer years;

    @PositiveOrZero
    private Integer totalDays;

    @PositiveOrZero
    private Integer takenDays;

    @NotBlank
    private String holidayStatus;


    public UUID getId() {
        return this.id;
    }
    public void setId(UUID id) {
        this.id = id;
    }
    public UUID getAccountId() {
        return this.accountId;
    }
    public void setAccountId(UUID accountId) {
        this.accountId = accountId;
    }
    public String getEmail() {
        return this.email;
    }
    public void setEmail(String email) {
        this.email = email;
```

```
    }
    public String getFirstName() {
        return this.firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    public String getLastName() {
        return this.lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
    public String getRole() {
        return this.role;
    }
    public void setRole(String role) {
        this.role = role;
    }
    public String getDepartment() {
        return this.department;
    }
    public void setDepartment(String department) {
        this.department = department;
    }
    public void setYears(Integer years) {this.years = years; }
    public Integer getYears() {return this.years;}
    public void setTotalDays(Integer totalDays) {this.totalDays = totalDays; }
    public Integer getTotalDays() {return this.totalDays;}
    public void setTakenDays(Integer takenDays) {this.takenDays = takenDays; }
    public Integer getTakenDays() {return this.takenDays;}
    public String getHolidayStatus() {
        return this.holidayStatus;
    }
    public void setHolidayStatus(String holidayStatus) {
        this.holidayStatus = holidayStatus;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
```

```
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.NotBlank;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;

/**
 * Request model for employees by date
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class FindEmployeesByDateRequest implements Serializable {
        private static final long serialVersionUID = 1L;

        @NotBlank
    private String date;

    public void setDate(String date) {
        this.date = date;
    }
    public String getDate() {return this.date; }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.NotBlank;

import com.holidaysystem.validation.ValueOfEnum;
import com.holidaysystem.enumeration.HolidayRequestStatusEnum;
```

```
import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Request model for holiday request
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class HolidayRequest implements Serializable {

    @NotBlank
    private UUID employeeId;

    @NotBlank
    @ValueOfEnum(enumClass = HolidayRequestStatusEnum.class)
    private String status;

    @NotBlank
    private String startDate;

    @NotBlank
    private String endDate;

    public UUID getEmployeeId() {
        return this.employeeId;
    }
    public void setEmployeeId(UUID employeeId) {
        this.employeeId = employeeId;
    }
    public String getStatus() {
        return this.status;
    }
    public void setStatus(String status) {
        this.status = status;
    }
    public String getStartDate() {
        return this.startDate;
    }
    public void setEndDate(String endDate) {
        this.endDate = endDate;
    }
    public String getEndDate() {
        return this.endDate;
    }
    public void setStartDate(String startDate) {
        this.startDate = startDate;
    }
```

```
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.PositiveOrZero;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Response model for holiday request
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class HolidayResponse implements Serializable {

        @NotBlank
    private UUID id;

    @NotBlank
    private UUID employeeId;

    @NotBlank
    private String status;

    private Integer requestedDays;

    @NotBlank
```

```
    private String startDate;

    @NotBlank
    private String endDate;

    private Integer years;

    private Integer totalDays;

    private Integer takenDays;

    public UUID getId() {
        return this.id;
    }
    public void setId(UUID id) {
        this.id = id;
    }
    public UUID getEmployeeId() {
        return this.employeeId;
    }
    public void setEmployeeId(UUID employeeId) {
        this.employeeId = employeeId;
    }
    public String getStatus() {
        return this.status;
    }
    public void setStatus(String status) {
        this.status = status;
    }
    public String getStartDate() {
        return this.startDate;
    }
    public void setStartDate(String startDate) {
        this.startDate = startDate;
    }
    public void setEndDate(String endDate) {
        this.endDate = endDate;
    }
    public String getEndDate() {
        return this.endDate;
    }
    public void setYears(Integer years) {this.years = years; }
    public Integer getYears() {return this.years;}
    public void setTotalDays(Integer totalDays) {this.totalDays = totalDays; }
    public Integer getTotalDays() {return this.totalDays;}
    public void setTakenDays(Integer takenDays) {this.takenDays = takenDays; }
    public Integer getTakenDays() {return this.takenDays;}
    public void setRequestedDays(Integer requestedDays) {this.requestedDays =
       requestedDays; }
    public Integer getRequestedDays() {return this.requestedDays;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
```

```
     eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.Email;
import javax.validation.constraints.NotBlank;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;

/**
 * Request model for login
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class LoginRequest implements Serializable {

    @Email
    private String email;

    @NotBlank
    private String password;

    public String getEmail() {
        return this.email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getPassword() {
        return this.password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
}
```

```
/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.PositiveOrZero;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;
import java.util.UUID;

/**
 * Request model for employee
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class NewEmployeeRequest implements Serializable {

    @NotBlank
    private String firstName;

    @NotBlank
    private String lastName;

    @NotBlank
    private UUID accountId;

    @NotBlank
    private String role;

    @NotBlank
    private String department;

    @PositiveOrZero
```

```
    private Integer years;


    public String getFirstName() {
        return this.firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    public String getLastName() {
        return this.lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
    public UUID getAccountId() {
        return this.accountId;
    }
    public void setAccountId(UUID accountId) {
        this.accountId = accountId;
    }
    public String getRole() {
        return this.role;
    }
    public void setRole(String role) {
        this.role = role;
    }
    public String getDepartment() {
        return this.department;
    }
    public void setDepartment(String department) {
        this.department = department;
    }
    public void setYears(Integer years) {this.years = years; }
    public Integer getYears() {return this.years;}
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
```

```
package com.holidaysystem.vo;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;

/**
 * Prioritized records request
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class PrioritizedRecordsRequest implements Serializable {

        private String date;

        private String requestStatus;

    public String getRequestStatus() {
        return this.requestStatus;
    }
    public void setRequestStatus(String requestStatus) {
        this.requestStatus = requestStatus;
    }
    public String getDate() {
        return this.date;
    }
    public void setDate(String date) {
        this.date = date;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;
```

```
import javax.validation.constraints.Email;
import javax.validation.constraints.NotBlank;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.io.Serializable;

/**
 * Request model for registration
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class RegistrationRequest implements Serializable {

    @Email
    private String email;

    @NotBlank
    private String password;

    @NotBlank
    private String authRole;

    public String getEmail() {
        return this.email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getPassword() {
        return this.password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public String getAuthRole() {
        return this.authRole;
    }
    public void setAuthRole(String authRole) {
        this.authRole = authRole;
    }
}

/*
 *     Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *     Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
```

```
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import java.io.Serializable;
import java.util.UUID;

import javax.validation.constraints.NotBlank;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

/**
 * Response model for request alert
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class RequestAlertResponse implements Serializable {

        @NotBlank
    private UUID id;

        @NotBlank
    private UUID requestId;

    @NotBlank
    private String date;

    public UUID getId() {
        return this.id;
    }
    public void setId(UUID id) {
        this.id = id;
    }
    public UUID getRequestId() {
        return this.requestId;
    }
    public void setRequestId(UUID requestId) {
        this.requestId = requestId;
    }
    public String getDate() {
        return this.date;
    }
```

```
    public void setDate(String date) {
        this.date = date;
    }
}

/*
 *      Copyright 2022-2022 Yauhen Bichel yb3129h@gre.ac.uk OR bichel.
    eugen@gmail.com
 *      Student Id 001185491
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *          http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package com.holidaysystem.vo;

import java.io.Serializable;
import java.util.UUID;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.NotNull;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlRootElement;

/**
 * Response model for request alert
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 *
 */
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class RequestValidationResponse implements Serializable {

        @NotBlank
    private UUID requestId;

        @NotNull
    private Boolean valid;

    public UUID getRequestId() {
        return this.requestId;
    }
    public void setRequestId(UUID requestId) {
        this.requestId = requestId;
```

```
    }
    public Boolean getValid() {
        return this.valid;
    }
    public void setValid(Boolean valid) {
        this.valid = valid;
    }
}


package com.holidaysystem.mapper;

import java.util.UUID;

import org.junit.jupiter.api.Assertions;
import org.junit.jupiter.api.Test;
import static org.junit.Assert.*;

import com.holidaysystem.common.RegistrationRequestBuilder;
import com.holidaysystem.entity.AccountEntity;
import com.holidaysystem.enumeration.AuthorizationRoleEnum;
import com.holidaysystem.mapper.AccountMapper;
import com.holidaysystem.vo.RegistrationRequest;

/**
 * Unit tests for AccountMapper class.
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 */
public class AccountMapperTest {

        @Test
        public void testToEntity() {
                //arrange
                UUID id = UUID.randomUUID();
                RegistrationRequest registrationRequest =
                                RegistrationRequestBuilder.builder()
                                .setEmail("test.test@gmail.com")
                                .setPassword("test pass !")
                                .setAuthRole(AuthorizationRoleEnum.ADMIN.name
                                    ())
                                .build();

                String hashedPassWithSalt = UUID.randomUUID().toString();
                AccountMapper mapper = new AccountMapper();

                //act
                AccountEntity actualEntity = mapper.toEntity(id,
                    registrationRequest, hashedPassWithSalt);

                //assert
                assertEquals(registrationRequest.getEmail(), actualEntity.
                    getEmail());
                assertEquals(hashedPassWithSalt, actualEntity.getPassword());
        }
```

}

] Unit Tests

```
package com.holidaysystem.mapper;

import java.util.UUID;

import org.junit.jupiter.api.Assertions;
import org.junit.jupiter.api.Test;
import static org.junit.Assert.*;

import com.holidaysystem.common.RegistrationRequestBuilder;
import com.holidaysystem.entity.AccountEntity;
import com.holidaysystem.enumeration.AuthorizationRoleEnum;
import com.holidaysystem.mapper.AccountMapper;
import com.holidaysystem.vo.RegistrationRequest;

/**
 * Unit tests for AccountMapper class.
 * @author yauhen bichel yb3129h@gre.ac.uk Student Id 001185491
 */
public class AccountMapperTest {

	@Test
	public void testToEntity() {
		//arrange
		UUID id = UUID.randomUUID();
		RegistrationRequest registrationRequest =
				RegistrationRequestBuilder.builder()
				.setEmail("test.test@gmail.com")
				.setPassword("test pass !")
				.setAuthRole(AuthorizationRoleEnum.ADMIN.name
				    ())
				.build();

		String hashedPassWithSalt = UUID.randomUUID().toString();
		AccountMapper mapper = new AccountMapper();

		//act
		AccountEntity actualEntity = mapper.toEntity(id,
		    registrationRequest, hashedPassWithSalt);

		//assert
		assertEquals(registrationRequest.getEmail(), actualEntity.
		    getEmail());
		assertEquals(hashedPassWithSalt, actualEntity.getPassword());
	}
}
```