1. CreatePersonSightingTest

```
package com.promineotech.person.controller;

import static org.assertj.core.api.Assertions.assertThat;

import java.time.LocalDate;

import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.boot.test.context.SpringBootTest.WebEnvironment;
import org.springframework.boot.test.web.client.TestRestTemplate;
import org.springframework.boot.test.web.server.LocalServerPort;
import org.springframework.test.context.ActiveProfiles;
import org.springframework.test.context.jdbc.Sql;
import org.springframework.test.context.jdbc.SqlConfig;
import org.springframework.test.jdbc.JdbcTestUtils;

import com.promineotech.person.entity.PersonSighting;

import org.springframework.http.HttpEntity;
import org.springframework.http.HttpHeaders;
import org.springframework.http.HttpMethod;
import org.springframework.http.HttpStatus;
import org.springframework.http.MediaType;
import org.springframework.http.ResponseEntity;

@SpringBootTest(webEnvironment = WebEnvironment.RANDOM_PORT)
@ActiveProfiles("test")
@Sql(scripts = {
   "classpath:flyway/migrations/V1.0__Person_Schema.sql",
   "classpath:flyway/migrations/V1.1__Person_Data.sql"},
   config = @SqlConfig(encoding = "utf-8"))
class CreatePersonSightingTest {

	@LocalServerPort
	private int serverPort;

	@Autowired
	private TestRestTemplate restTemplate;

	@Test
	void testCreatePersonSightingReturnsSuccess201() {
		//Given: an personSighting as JSON
		String body = createPersonSightingBody();
		String uri = String.format("http://localhost:%d/personSighting", serverPort);

		HttpHeaders headers = new HttpHeaders();
		headers.setContentType(MediaType.APPLICATION_JSON);

		HttpEntity<String> bodyEntity = new HttpEntity<>(body, headers);

		//When: the personSighting is sent
		ResponseEntity<PersonSighting> response = restTemplate.exchange(uri,
				HttpMethod.POST, bodyEntity, PersonSighting.class);
			//Then:a 201 status is returned
			assertThat(response.getStatusCode()).isEqualTo(HttpStatus.CREATED);

			//And: the returned personSighting is correct
			assertThat(response.getBody()).isNotNull();

			PersonSighting personSighting = response.getBody();
			assertThat(personSighting.getSighting().getSightingId()).isEqualTo("YANG_BO");
//
		assertThat(personSighting.getSighting().getSightingDate()).isEqualTo(LocalDate.parse("2022-
3-15"));
//
		assertThat(personSighting.getSighting().getSightingProvinceId()).isEqualTo("GUIZHOU");


			assertThat(personSighting.getPerson().getPersonId()).isEqualTo("YANG_BO");
//			assertThat(personSighting.getPerson().getFamilyName()).isEqualTo("YANG");
//			assertThat(personSighting.getPerson().getGivenName()).isEqualTo("BO");
//
		assertThat(personSighting.getPerson().getBirthday()).isEqualTo(LocalDate.parse("2018-12-
01"));
//			assertThat(personSighting.getPerson().getGender()).isEqualTo("male");
//
		assertThat(personSighting.getPerson().getMissingDate()).isEqualTo(LocalDate.parse("2022-1-
28"));
//			assertThat(personSighting.getPerson().getHomeProvinceId()).isEqualTo("HENAN");

		}
		protected String createPersonSightingBody() {
		// @formatter:off
			return "{\n"
			+ "  \"sighting\":\"YANG_BO\",\n"
//			+ "  \"sightingDate\":\"2022-3-15\",\n"
//		  + "  \"sightingProvince\":\"GUIZHOU\",\n"
		  + "  \"person\":\"YANG_BO\"\n"
//		  + "  \"familyName\":\"YANG\",\n"
//		  + "  \"givenName\":\"BO\",\n"
//		  + "  \"birthday\":\"2018-12-01\",\n"
//		  + "  \"gender\":\"male\",\n"
//		  + "  \"missingDate\":\"2022-1-28\",\n"
//		  + "  \"homeProvince\":\"HENAN\"\n"
		  + "}";
		//@formatter:on
		}

}
```

2. PersonSightingController

```
package com.promineotech.person.controller;

import javax.validation.Valid;

import org.springframework.http.HttpStatus;
import org.springframework.validation.annotation.Validated;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseStatus;

import com.promineotech.person.entity.PersonSighting;
import com.promineotech.person.entity.PersonSightingRequest;

import io.swagger.v3.oas.annotations.OpenAPIDefinition;
import io.swagger.v3.oas.annotations.Operation;
import io.swagger.v3.oas.annotations.info.Info;
import io.swagger.v3.oas.annotations.servers.Server;
import io.swagger.v3.oas.annotations.responses.ApiResponse;
import io.swagger.v3.oas.annotations.media.Content;
import io.swagger.v3.oas.annotations.media.Schema;
import io.swagger.v3.oas.annotations.Parameter;

@RequestMapping("/personSighting")
@OpenAPIDefinition(info = @Info(title = "PersonSighting service"), servers = {
@Server(url = "http://localhost:8080",description = "local server.")})
@Validated
public interface PersonSightingController {
//@formatter:off
@Operation(
summary = "Create a PersonSighting",
description = "returns the created PersonSighting",
responses = {
@ApiResponse(
responseCode = "201",
description = "The created PersonSighting is returned",
content = @Content(mediaType = "application/json",
schema =@Schema(implementation = PersonSighting.class))),
@ApiResponse(
responseCode = "400",
description = "The request parameters are invalid",
content = @Content(mediaType = "application/json")
),
@ApiResponse(
responseCode = "404",
description = "A PersonSighting component was not found with the input criteria",
content = @Content(mediaType = "application/json")
),
@ApiResponse(
responseCode = "500",
description = "An unplanned error occurred",
content = @Content(mediaType = "application/json")
)
},
parameters = {
@Parameter(
name = "personSightingRequest",
required = true,
description = "The personSighting as JSON"),
}
)
@PostMapping
@ResponseStatus(code = HttpStatus.CREATED)
PersonSighting createPersonSighting(
@Valid@RequestBody PersonSightingRequest personSightingRequest
);
//@formatter:on

}
```

3. DefaultPersonSightingController

```
package com.promineotech.person.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.RestController;

import com.promineotech.person.entity.PersonSighting;
import com.promineotech.person.entity.PersonSightingRequest;
import com.promineotech.person.service.PersonSightingService;

import lombok.extern.slf4j.Slf4j;

@RestController
@Slf4j
public class DefaultPersonSightingController implements PersonSightingController {

@Autowired
private PersonSightingService personSightingService;
@Override
public PersonSighting createPersonSighting(PersonSightingRequest personSightingRequest) {
log.debug("PersonSighting={}", personSightingRequest);
return personSightingService.createPersonSighting(personSightingRequest);
}
}
```

4. PersonSightingService

```
package com.promineotech.person.service;

import com.promineotech.person.entity.PersonSighting;
import com.promineotech.person.entity.PersonSightingRequest;

public interface PersonSightingService {

PersonSighting createPersonSighting(PersonSightingRequest personSightingRequest);

}
```

5. DefaultPersonSightingService

```
package com.promineotech.person.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.promineotech.person.dao.PersonSightingDao;
import com.promineotech.person.entity.Person;
import com.promineotech.person.entity.PersonSighting;
import com.promineotech.person.entity.PersonSightingRequest;

import com.promineotech.person.entity.Sighting;

@Service
public class DefaultPersonSightingService implements PersonSightingService {

@Autowired
private PersonSightingDao personSightingDao;
@Transactional
@Override
public PersonSighting createPersonSighting(PersonSightingRequest personSightingRequest) {
Sighting sighting = getSighting(personSightingRequest);
Person person = getPerson(personSightingRequest);
return personSightingDao.savePersonSighting(sighting, person);
}

/**
*
* @param personSightingRequest
* @return
*/
private Person getPerson(PersonSightingRequest personSightingRequest) {
return personSightingDao.fetchPerson(personSightingRequest.getPerson());
}

/**
*
* @param personSightingRequest
* @return
*/
private Sighting getSighting(PersonSightingRequest personSightingRequest) {
return personSightingDao.fetchSighting(personSightingRequest.getSighting());
}

}
```

6. PersonSightingDao

```
package com.promineotech.person.dao;

import com.promineotech.person.entity.Person;
import com.promineotech.person.entity.PersonSighting;
import com.promineotech.person.entity.Sighting;

public interface PersonSightingDao {
Person fetchPerson(String personId);
Sighting fetchSighting(String sightingId);
PersonSighting savePersonSighting(Sighting sighting, Person person);
}
```

7. DefaultPersonSightingDao

```
package com.promineotech.person.dao;
```

```
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.HashMap;

import java.util.Map;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.ResultSetExtractor;
import org.springframework.jdbc.core.namedparam.MapSqlParameterSource;
import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;
import org.springframework.jdbc.support.GeneratedKeyHolder;
import org.springframework.jdbc.support.KeyHolder;
import org.springframework.stereotype.Component;

import com.promineotech.person.entity.Person;
import com.promineotech.person.entity.PersonSighting;

import com.promineotech.person.entity.Sighting;

@Component
public class DefaultPersonSightingDao implements PersonSightingDao {

@Autowired
private NamedParameterJdbcTemplate jdbcTemplate;
@Override
public PersonSighting savePersonSighting(Sighting sighting, Person person) {
SqlParams params = generateInsertSql(sighting,person);
KeyHolder keyHolder = new GeneratedKeyHolder();
jdbcTemplate.update(params.sql, params.source,keyHolder);
Long personSightingPk = keyHolder.getKey().longValue();
//@formatter:off
return PersonSighting.builder()
.personSightingPK(personSightingPk)
.sighting(sighting)
.person(person)
.build();
//@formatter:on
};
/**
*
* @param person
* @param sighting
* @param
* @return
*/
private SqlParams generateInsertSql(Sighting sighting, Person person) {
// @formatter:off
String sql = ""
+ "INSERT INTO person_sighting ("
+ "person_sighting_id, sighting_fk, person_fk"
+ ") VALUES ("
+ ":person_sighting_id, :sighting_fk, :person_fk"
+ ")";
// @formatter:on
SqlParams params = new SqlParams();
System.out.println("in generateInsertSql");
System.out.println(person.getPersonPK());
System.out.println(sighting.getSightingPK());
params.sql = sql;
params.source.addValue("sighting_fk", sighting.getSightingPK());
params.source.addValue("person_fk", person.getPersonPK());
params.source.addValue("person_sighting_id", "Hello");
return params;
}
/**
*
*/
@Override
public Person fetchPerson(String personId) {
// @formatter:off
String sql = ""
+ "SELECT * "
+ "FROM person "
+ "WHERE person_id = :person_id ";

// @formatter:on

Map<String, Object> params = new HashMap<>();
params.put("person_id", personId);

Person tmp_person = jdbcTemplate.query(sql, params, new PersonResultSetExtractor());
tmp_person.setPersonPK((long)1);
return tmp_person;
}

/**
*
*/
@Override
public Sighting fetchSighting(String sightingId) {
// @formatter:off
String sql = ""
+ "SELECT * "
+ "FROM sighting "
+ "WHERE sighting_id = :sighting_id ";

// @formatter:on

Map<String, Object> params = new HashMap<>();
params.put("sighting_id", sightingId);
Sighting tmp_sighting = jdbcTemplate.query(sql, params, new SightingResultSetExtractor());
tmp_sighting.setSightingPK((long) 2);
return tmp_sighting;
}



/**
*
* @author Promineo
*
*/
class PersonResultSetExtractor implements ResultSetExtractor<Person> {
@Override
```

```
public Person extractData(ResultSet rs) throws SQLException {
rs.next();

// @formatter:off
return Person.builder()
.personId(rs.getString("person_id"))
.familyName(rs.getString("family_name"))
.givenName(rs.getString("given_name"))
// .birthday(rs.getDate("birthday").toLocalDate())
.gender(rs.getString("gender"))
// .missingDate(rs.getDate("missing_date").toLocalDate())
.homeProvinceId(rs.getString("Home_province_id"))
.build();
// @formatter:on
}
}

/**
*
* @author Promineo
*
*/
class SightingResultSetExtractor implements ResultSetExtractor<Sighting> {
@Override
public Sighting extractData(ResultSet rs) throws SQLException {
rs.next();

// @formatter:off
return Sighting.builder()
.sightingId(rs.getString("sighting_id"))
// .sightingDate(rs.getDate("sighting_date").toLocalDate())
.sightingProvinceId(rs.getString("sighting_province_id"))
.build();
// @formatter:on
}
}


class SqlParams {
String sql;
MapSqlParameterSource source = new MapSqlParameterSource();
}


}
```