1. Application

```
package com.promineotech.person;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class PersonSighting {

	public static void main(String[] args) {
		SpringApplication.run(PersonSighting.class, args);
	}
}
```

2. FetchPersonTest

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

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

import java.time.LocalDate;
import java.util.LinkedList;
import java.util.List;

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.core.ParameterizedTypeReference;
import org.springframework.http.HttpMethod;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.test.context.ActiveProfiles;
import org.springframework.test.context.jdbc.Sql;
import org.springframework.test.context.jdbc.SqlConfig;

import com.promineotech.person.entity.Person;

@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 FetchPersonTest {

	@Autowired
	private TestRestTemplate restTemplate;

	@LocalServerPort
	private int serverPort;
```

```
	protected List<Person> buildExpected() {
		List<Person> list = new LinkedList<>();
		list.add(Person.builder()
					.personId("YANG_BO")
					.familyName("Yang")
					.givenName("Bo")
					.birthday(LocalDate.parse("2018-12-01"))
					.gender("male")
					.missingDate(LocalDate.parse("2022-01-28"))
					.homeProvinceId("HENAN")
					.build());

		return list;
	}

	@Test
	void testThatPersonAreReturnedWhenAValidNameAndGenderAreSupplied() {
		// Given: a valid model, trim and URI
		String personId = "YANG_BO";
		String homeProvinceId = "HENAN";
		String gender = "male";
		String uri =

	String.format("http://localhost:%d/person?personId=%s&homeProvinceId=%s&gender=%s",
serverPort,
					personId, homeProvinceId, gender);
	  System.out.println(uri);
	// When: a connection is made to the URI

    ResponseEntity<List<Person>> response =
					restTemplate.exchange(uri, HttpMethod.GET, null, new
ParameterizedTypeReference<>(){});
	// Then: a success (OK - 200) status code is returned
		assertThat(response.getStatusCode()).isEqualTo(HttpStatus.OK);

	// And: the actual list returned is the same as the expected list.
		List<Person> actual = response.getBody();
		List<Person> expected = buildExpected();

		assertThat(actual).isEqualTo(expected);
	}


}
```

3. PersonLostController

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

import java.util.List;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
```

```
import org.springframework.web.bind.annotation.ResponseStatus;

import com.promineotech.person.entity.Person;

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


@RequestMapping("/person")
@OpenAPIDefinition(info = @Info(title = "Person information"), servers = {
@Server(url = "http://localhost:8080",description = "local server.")})
public interface PersonLostController {
	//@formatter:off
		@Operation(
				summary = "Returns a list of person information",
				description = "returns a list of person information given personId
and homeProvinceId and gender",
				responses = {
						@ApiResponse(
								responseCode = "200",
								description = "A list of person
information is returned",
								content =
@Content(mediaType = "application/json",
								schema
=@Schema(implementation = Person.class))),
						@ApiResponse(
								responseCode = "400",
								description = "The request
parameters are invalid",
								content =
@Content(mediaType = "application/json")
								),
						@ApiResponse(
								responseCode = "404",
								description = "No Person was
found with the input",
								content =
@Content(mediaType = "application/json")
								),
						@ApiResponse(
								responseCode = "500",
								description = "An unplanned
error occurred",
								content =
@Content(mediaType = "application/json")
								)
				},
				parameters = {
						@Parameter(
								name = "personId",
```

```
                                                        allowEmptyValue = false,
                                                        required = false,
                                                        description = "The personId is
(i.e,,'YANG_BO')"),
                                                @Parameter(
                                                        name = "homeProvinceId",
                                                        allowEmptyValue = false,
                                                        description = "The
homeProvinceId is (i.e,,'HENAN')"),
                                                @Parameter(
                                                        name = "gender",
                                                        allowEmptyValue = false,
                                                        description = "The gender is
(i.e,,'male')")
                                        }
                                        )
                        @GetMapping
                        @ResponseStatus(code = HttpStatus.OK)
                        List<Person> fetchPerson(
                                        @RequestParam String personId,
                            @RequestParam String homeProvinceId,
                            @RequestParam String gender
                            );
                //@formatter:on
}
```

4. DefaultPersonLostController

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

import java.util.List;

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

import com.promineotech.person.entity.Person;
import com.promineotech.person.service.PersonLostService;

import lombok.extern.slf4j.Slf4j;

@RestController
@Slf4j
public class DefaultPersonLostController implements PersonLostController {

        @Autowired
        private PersonLostService personSightingService;

        @Override
        public List<Person> fetchPerson(String personId, String homeProvinceId, String gender) {
                log.debug("personId = {}, provinceId = {}, gender = {}", personId,
                                homeProvinceId, gender);
                return personSightingService.fetchPerson(personId, homeProvinceId, gender);

        }

}
```

5. PersonLostService

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

import java.util.List;

import com.promineotech.person.entity.Person;

public interface PersonLostService {

	List<Person> fetchPerson(String personId, String homeProvinceId, String gender);

}
```

6. DefaultPersonLostService

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

import java.util.List;

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

import com.promineotech.person.dao.PersonLostDao;
import com.promineotech.person.entity.Person;

import lombok.extern.slf4j.Slf4j;

@Service
@Slf4j
public class DefaultPersonLostService implements PersonLostService {
@Autowired
private PersonLostDao personSightingDao;
@Override
public List<Person> fetchPerson(String personId, String homeProvinceId, String gender) {
log.debug("The fetchPerson method was called with personId = {}," +
"and provinceId = {}," + "and gender = {}", personId,
homeProvinceId, gender);

return personSightingDao.fetchPerson(personId, homeProvinceId, gender);
}

}
```

7. PersonLostDao

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

import java.util.List;

import com.promineotech.person.entity.Person;

public interface PersonLostDao {
	List<Person> fetchPerson(String personId, String homeProvinceId, String gender);
}
```

8. DefaultPersonLostDao

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

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;
import org.springframework.stereotype.Service;

import com.promineotech.person.entity.Person;

import lombok.extern.slf4j.Slf4j;

@Service
@Slf4j
public class DefaultPersonLostDao implements PersonLostDao {

@Autowired
private NamedParameterJdbcTemplate jdbcTemplate;
@Override
public List<Person> fetchPerson(String personId, String homeProvinceId, String gender) {
log.debug("DAO: personId = {}, provinceId = {}, gender = {}", personId,
homeProvinceId, gender);
// @formatter: off
String sql = ""
+ "SELECT * "
+ "FROM person "
+ "WHERE person_id = :person_id AND Home_province_id = :Home_province_id AND gender
= :gender";
// @formatter: on
Map<String, Object> params = new HashMap<>();
params.put("person_id", personId);
params.put("Home_province_id", homeProvinceId);
params.put("gender", gender);
return jdbcTemplate.query(sql, params, new RowMapper<>(){

@Override
public Person mapRow(ResultSet rs, int rowNum) throws SQLException {
//@formatter:off
return Person.builder()
.personPK(rs.getLong("person_Pk"))
.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
```

```
}});
}

}
```