

Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №7
По дисциплине «ПИС»
за 5-й семестр

Выполнил:
студент 3 курса
группы ПО-3 (1)
Лущ М. Г.

Проверил:
Лаврущук А.И.

Брест, 2021

Вариант 4

Цель работы: Познакомиться с практической реализацией принципа инверсии зависимостей, а также протоколами межсервисного взаимодействия.

Задание для выполнения:

Реализуйте механизмы хранения (persistence) для ранее разработанного приложения – технологию, фреймворк, ORM выберите сами. Это могут быть реляционная БД, NoSQL, InMemory, файловая система, Redis и т.д. Реализуйте минимум два механизма доставки – способов вызова функций вашего приложения. Например, REST и командная строка. Отдельно реализуйте микросервис, который не будет выполнять никакой полезной работы, кроме вызова какой-либо функции вашего приложения по протоколу gRPC (третий способ доставки для вашего приложения), используйте Google Protobuf. Большим плюсом станет, если данный микросервис будет реализован не на PHP:

Тема: Учёт военнообязанных.

Ход работы

Код Api:

```
<?php

header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");

include("../DatabaseContext.php");
include("../Repositories/MilitaryReservistRepository.php");
include("../Repositories/ReservistRepository.php");
include("../Application/ReservistRegistrationService.php");
include("../Entities/Reservist.php");

$connection = DatabaseContext::getInstance()->getConnection();
$militaryReservistRepository = new MilitaryReservistRepository($connection);
$reservistRepository = new ReservistRepository($connection);

$service = new ReservistRegistrationService($militaryReservistRepository, $reservistRepository);

$result = $service->getAllReservists();

if (count($result) == 0)
{
    http_response_code(404);
    echo json_encode(array('message' => 'reservists not found'));
}
```

```

else
{
    http_response_code(200);

    $response = array();
    foreach ($result as $value)
    {
        $responseItem = array(
            "id" => $value->getId(),
            "firstName" => $value->getFirstName(),
            "lastName" => $value->getLastName(),
            "birthDate" => $value->getBirthDate()->format('Y-m-d H:i:s'),
        );

        array_push($response, $responseItem);
    }

    echo json_encode($response);
}

```

Отдельный сервис взаимодействующий по протоколу gRPC:

Серверная часть:

```

const PROTO_PATH = 'definitions.proto';

const grpc = require('@grpc/grpc-js');
const protoLoader = require('@grpc/proto-loader');

const packageDefinition = protoLoader.loadSync(
    PROTO_PATH,
    {
        keepCase: true,
        longs: String,
        enums: String,
        defaults: true,
        oneofs: true
    });
}

var hello_proto = grpc.loadPackageDefinition(packageDefinition);

function Reservist(call, callback) {
    var fetch = require('node-fetch');

    fetch('http://localhost:8001/Api/Reservists.php')
        .then(res => {
            if(res.ok) return res
            else throw new Error(`The HTTP status of the reponse: ${res.status} (${res.statusText})`)
        })
}

```

```

        .then(res => res.json())
        .then(json => {
            console.log(json);
            console.log(call.request.id);

            const data = json;
            const query = data.filter(el => el.id === call.request.id)
            callback(null, query[0]);
        })
    }

function main() {
    const server = new grpc.Server();
    server.addService(hello_proto.ReservistService.service, {Reservist: Reservist});
    server.bindAsync('0.0.0.0:50051', grpc.ServerCredentials.createInsecure(), () => {
        server.start();
    });
}

main();

```

Клиентская часть:

```

const PROTO_PATH = 'definitions.proto';

var parseArgs = require('minimist');
var grpc = require('@grpc/grpc-js');
var protoLoader = require('@grpc/proto-loader');

var packageDefinition = protoLoader.loadSync(
    PROTO_PATH,
    {keepCase: true,
     longs: String,
     enums: String,
     defaults: true,
     oneofs: true
    });

var hello_proto = grpc.loadPackageDefinition(packageDefinition);

function main() {
    var argv = parseArgs(process.argv.slice(1));
    const customerID = argv['Id'];

    var client = new hello_proto.ReservistService('localhost:50051',
        grpc.credentials.createInsecure());

    client.Reservist({id: customerID}, function(err, response) {
        console.log('Reservist:', response);
    });
}

```

```
});
```

```
}
```

```
main();
```

Результат работы API:

The screenshot shows a JSON viewer interface with the URL `localhost:8001/Api/Reservists.php`. The JSON data is displayed under the heading "JSON". The data consists of two arrays: "0:" and "1:". Array "0:" contains one object with fields: id: 1, firstName: "test", lastName: "test", birthDate: "2021-05-04 23:48:22". Array "1:" contains one object with fields: id: 2, firstName: "34532", lastName: "3425", birthDate: "2021-05-05 00:00:00". Other tabs like "Raw Data" and "Headers" are visible at the top.

```
[{"id": 1, "firstName": "test", "lastName": "test", "birthDate": "2021-05-04 23:48:22"}, {"id": 2, "firstName": "34532", "lastName": "3425", "birthDate": "2021-05-05 00:00:00"}]
```

Результат работы в консоли:

```
root@8fe14eed5f3b:/var/www/html# ls
Api      ApplicationTests  Controllers    Entities    Views     composer.json  css   js   vendor
Application  Console        DatabaseContext.php  Repositories bootstrap.php  composer.lock  index.php  phpunit.xml
root@8fe14eed5f3b:/var/www/html# cd Api/
root@8fe14eed5f3b:/var/www/html/Api# php -f Reservists.php
[{"id":1,"firstName":"test","lastName":"test","birthDate":"2021-05-05 00:19:18"}, {"id":2,"firstName":"34532","lastName":"3425","birthDate":"2021-05-05 00:00:00"}]root@8fe14eed5f3b:/
var/www/html/Api#
```

Результат работы gRPC:

```
architect-prog@architectprog-VirtualBox:~/Desktop/node_grpc/server$ node server.js
[
  {
    id: 1,
    firstName: 'test',
    lastName: 'test',
    birthDate: '2021-05-05 00:17:15'
  },
  {
    id: 2,
    firstName: '34532',
    lastName: '3425',
    birthDate: '2021-05-05 00:00:00'
  }
]
```

```
architect-prog@architectprog-VirtualBox:~/Desktop/node_grpc/client$ node client --Id 1
Reservist: [
  {
    id: 1,
    firstName: 'test',
    lastName: 'test',
    birthDate: '2021-05-05 00:17:15'
  }
]
architect-prog@architectprog-VirtualBox:~/Desktop/node_grpc/client$
```