**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**

**ІМЕНІ ІГОРЯ СІКОРСЬКОГО”**

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота №2**

**з дисципліни** «Бази даних»

**на тему:** Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL”

**спеціальність**: 121 – Програмна інженерія

**Виконав**

студент II курсу

групи КП-91

Бабак Артем Андрiйович

**Перевірив**

викладач

Петрашенко Андрій Васильович

Київ – 2020

**Завдання:**

- Реалізувати функції внесення, редагування та вилучення даних у таблицях бази даних, створених у лабораторній роботі №1, засобами консольного інтерфейсу.

- Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі.

- Забезпечити реалізацію пошуку за декількома атрибутами з двох та більше сутностей одночасно: для числових атрибутів – у рамках діапазону, для рядкових – як шаблон функції LIKE оператора SELECT SQL, для логічного типу – значення True/False, для дат – у рамках діапазону дат.

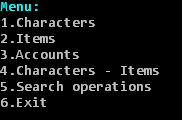
- Програмний код виконати згідно шаблону MVC (модель-подання-контролер).

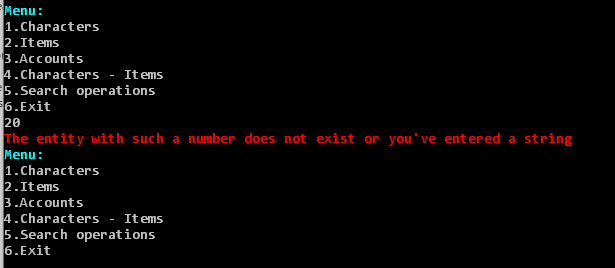
**Репозиторій:**

<https://github.com/tiwatit/Database>

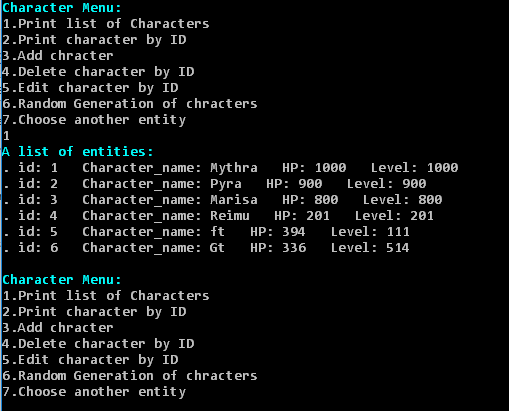
**Пункт №1**

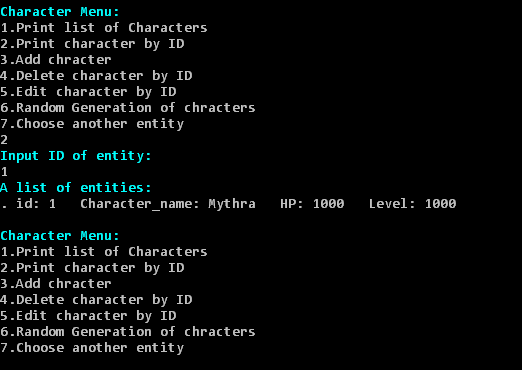
1. **Вибірсутності**

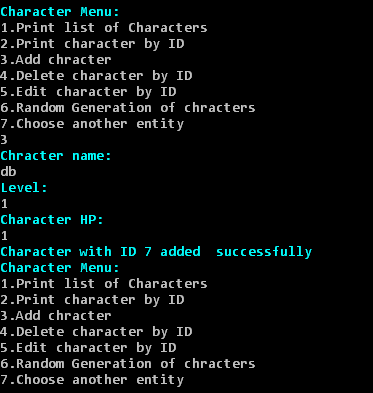


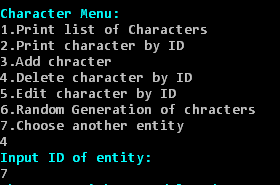


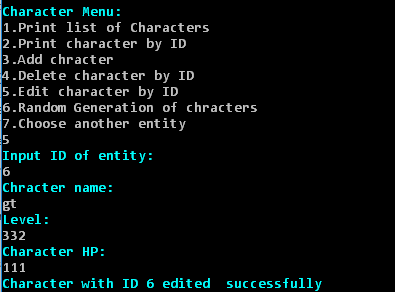
1. **Сутність «Character»**

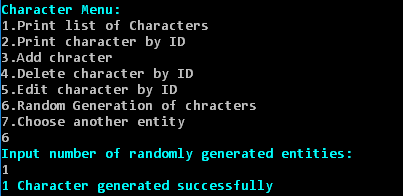




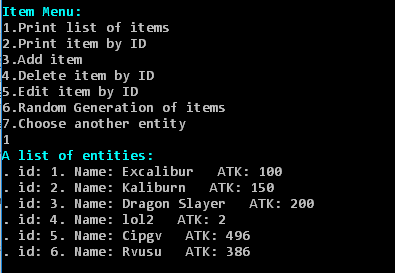


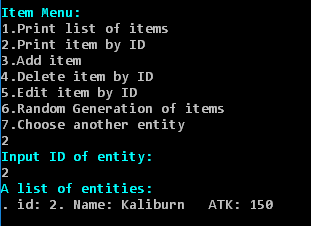


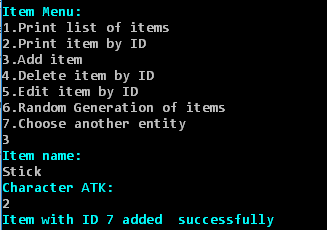


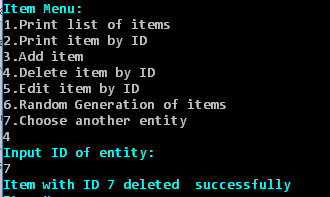


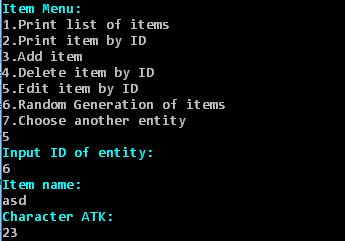
1. **Сутність *«*Item*»***

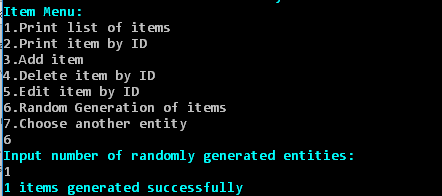




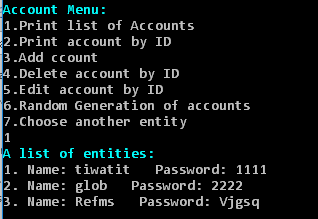


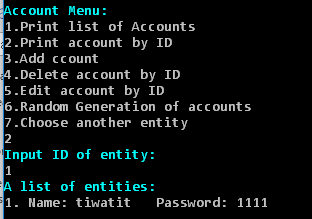


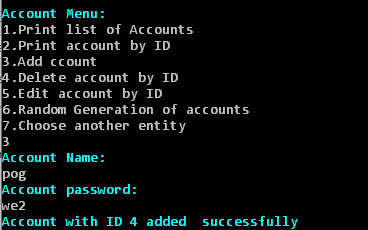


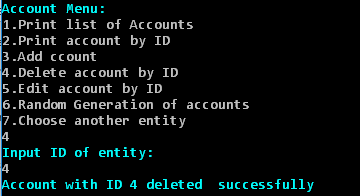


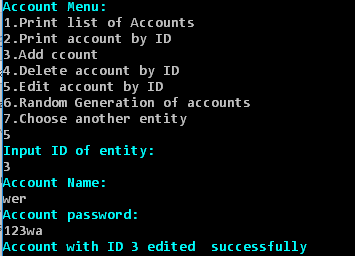
1. **Сутність *«*Account*»***

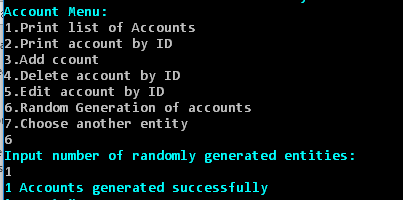




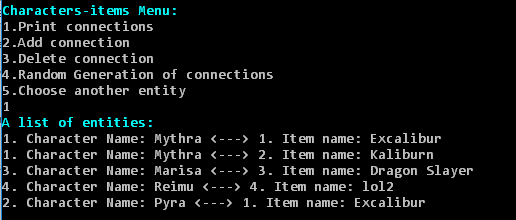


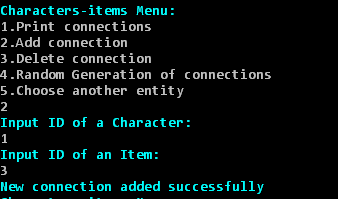


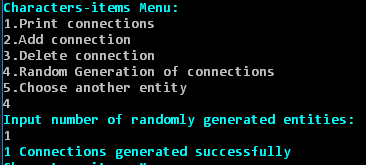


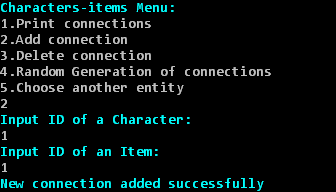


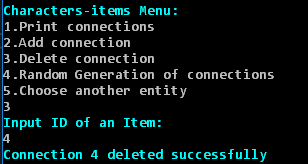
1. **Зв’язки між сутностями «Character» - «Item»**

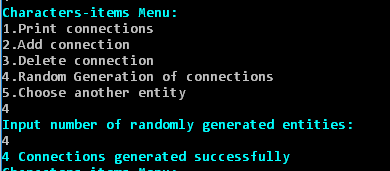






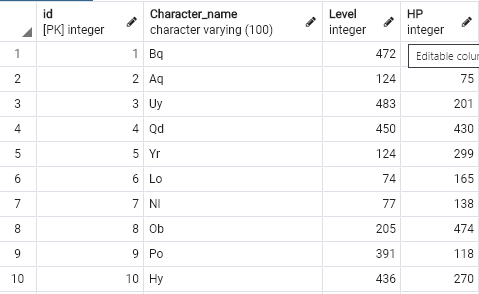






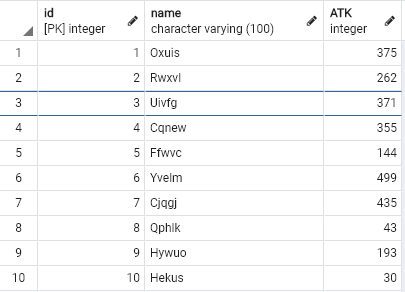
***Пункт №2***

1. Randomly generated Characters





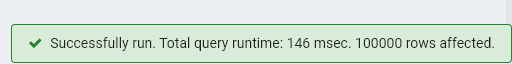
1. Randomly generated Items



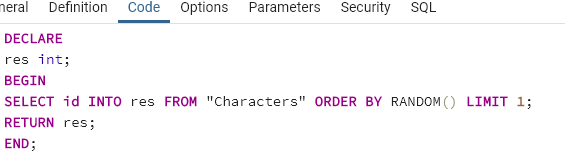


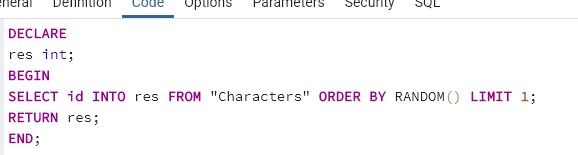
1. Randomly generated Accounts

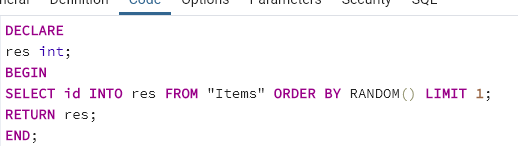


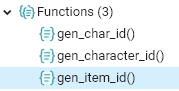


Для генерації зовнішніх ключів було прописано функції всередині самої бази даних:









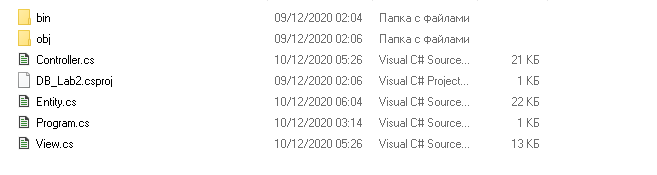
Генерація інших даних здійснена SQL-запитом у коді програми:

|  |
| --- |
| public void character\_generation(int num)  {  using var cmd = new NpgsqlCommand("INSERT INTO \"Characters\" (\"Character\_name\", \"HP\", \"Level\") SELECT chr(trunc(65 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) , trunc(random() \* 500 + 20), trunc(random() \* 500 + 20) FROM generate\_series(1, @num)", db);  cmd.Parameters.AddWithValue("@num", num);  cmd.ExecuteNonQuery();  } |
| public void Item\_generation(int num)  {  using var cmd = new NpgsqlCommand("INSERT INTO \"Items\" (\"name\", \"ATK\") SELECT chr(trunc(65 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int), trunc(random() \* 500 + 20) FROM generate\_series(1, @num)", db);  cmd.Parameters.AddWithValue("@num", num);  cmd.ExecuteNonQuery();  } |
| public void acc\_generation(int num)  {  using var cmd = new NpgsqlCommand("INSERT INTO \"accounts\" (\"name\", \"pword\") SELECT chr(trunc(65 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int), chr(trunc(65 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) FROM generate\_series(1, @num)", db);  cmd.Parameters.AddWithValue("@num", num);  cmd.ExecuteNonQuery();  } |
| public void acc\_item\_generation(int num)  {  using var cmd = new NpgsqlCommand("INSERT INTO \"Characters-Items\" (\"char\_id\", \"item\_id\") SELECT gen\_char\_id(), gen\_item\_id() FROM generate\_series(1, @num)", db);  cmd.Parameters.AddWithValue("@num", num);  cmd.ExecuteNonQuery();  } |

***Пункт №3***

|  |
| --- |
| SELECT \"Characters\".\"id\" AS \"Characters-Items.char\_id\", \"Characters\".\"Character\_name\", \"Characters\".\"Level\", \"Characters\".\"HP\", \"Characters-Items\".\"item\_id\", \"Items\".\"name\", \"Items\".\"ATK\" from \"Characters\" join \"Characters-Items\" on (\"Characters\".\"id\" = \"Characters-Items\".\"char\_id\")join \"Items\" on (\"Characters-Items\".\"item\_id\" = \"Items\".\"id\") WHERE \"Characters\".\"Level\" BETWEEN @s\_lvl AND @e\_lvl AND \"Characters\".\"id\" BETWEEN @s\_id AND @e\_id AND \"Items\".\"ATK\" BETWEEN @s\_ATK AND @e\_ATK |
| SELECT \"Characters\".\"id\" AS \"Characters-Items.char\_id\", \"Characters\".\"Character\_name\", \"Characters\".\"Level\", \"Characters\".\"HP\", \"Characters-Items\".\"item\_id\", \"Items\".\"name\", \"Items\".\"ATK\" from \"Characters\" join \"Characters-Items\" on (\"Characters\".\"id\" = \"Characters-Items\".\"char\_id\")join \"Items\" on (\"Characters-Items\".\"item\_id\" = \"Items\".\"id\") WHERE \"Characters\".\"Level\" BETWEEN @e\_lvl AND @s\_lvl AND \"Items\".\"name\" like '%" + i\_name + "%' AND \"Characters\".\"Character\_name\" like '%"+c\_name+"%' |
| "SELECT \"Characters\".\"id\" AS \"Characters-Items.char\_id\", \"Characters\".\"Character\_name\", \"Characters\".\"Level\", \"Characters\".\"HP\", \"Characters-Items\".\"item\_id\", \"Items\".\"name\", \"Items\".\"ATK\" from \"Characters\" join \"Characters-Items\" on (\"Characters\".\"id\" = \"Characters-Items\".\"char\_id\")join \"Items\" on (\"Characters-Items\".\"item\_id\" = \"Items\".\"id\") WHERE \"Characters\".\"Level\" BETWEEN @e\_hp AND @s\_hp AND \"Items\".\"name\" like '%" + i\_name + "%' AND \"Characters\".\"Character\_name\" like '%" + c\_name + "%'" |

***Пункт №4***



|  |
| --- |
| ***Program.cs***  using System;  using lab2.MVC;  namespace lab2  {  class Program  {  static void Main(string[] args)  {  Controller m = new Controller();  while (1 == 1)  {  if (m.entity\_menu() == 1)  {  break;  }  }  }  }  }  s |
| ***Entity.cs***  using System;  using System.Diagnostics;  using Npgsql;  namespace lab2.MVC  {  class Model  {  private NpgsqlConnection db= new NpgsqlConnection("Host = localhost; Username = postgres; Password = babak6832; Database = postgres");  public Model()  {  if (db == null)  {  NpgsqlConnection db = new NpgsqlConnection("Host = localhost; Username = postgres; Password = babak6832; Database = postgres");  db.Open();  using var cmd = new NpgsqlCommand("SELECT version()", db);  var version = cmd.ExecuteScalar().ToString();  Console.WriteLine($"PostgreSQL version: {version}");  }  else  {  db.Open();  using var cmd = new NpgsqlCommand("SELECT version()", db);  var version = cmd.ExecuteScalar().ToString();  Console.WriteLine($"PostgreSQL version: {version}");  }  }  #region Characters  public string character\_print()  {  using var cmd = new NpgsqlCommand("SELECT \* from \"Characters\" ORDER BY \"id\"", db);  using NpgsqlDataReader rdr = cmd.ExecuteReader();  string characters = "";  while (rdr.Read())  {  characters += ". id: ";  characters += rdr.GetInt32(0);  if (characters.Length == 0)  {  break;  }  characters += " Character\_name: ";  characters += rdr.GetString(1);  characters += " HP: ";  characters += rdr.GetInt32(2);  characters += " Level: ";  characters += rdr.GetInt32(3);  characters += "\n";  }  return characters;  }  public string character\_get\_by\_id(int id)  {  using var cmd = new NpgsqlCommand("SELECT \* FROM \"Characters\" WHERE \"id\"=" + id+"", db);  using NpgsqlDataReader rdr = cmd.ExecuteReader();  string characters = "";  while (rdr.Read())  {  characters += ". id: ";  characters += rdr.GetInt32(0);  if (characters.Length == 0)  {  break;  }  characters += " Character\_name: ";  characters += rdr.GetString(1);  characters += " HP: ";  characters += rdr.GetInt32(2);  characters += " Level: ";  characters += rdr.GetInt32(3);  characters += "\n";  }  return characters;  }  public int character\_add(string c\_name, int c\_HP, int c\_Level)  {    using var cmd = new NpgsqlCommand("INSERT INTO \"Characters\"(\"Character\_name\", \"HP\", \"Level\") VALUES(@name, @HP, @Level)", db);  cmd.Parameters.AddWithValue("name", c\_name);  cmd.Parameters.AddWithValue("HP", c\_HP);  cmd.Parameters.AddWithValue("Level", c\_Level);  cmd.Prepare();  cmd.ExecuteNonQuery();  using var cmd2 = new NpgsqlCommand("SELECT \"id\" FROM \"Characters\" WHERE id = (SELECT MAX(\"id\") from \"Characters\")", db);  using NpgsqlDataReader rdr = cmd2.ExecuteReader();  int new\_id = 0;  while (rdr.Read())  {  new\_id = rdr.GetInt32(0);  }  return new\_id;  }  public void character\_delete(int c\_id)  {  using var cmd2 = new NpgsqlCommand("DELETE from \"Characters\" WHERE \"id\"= " + c\_id, db);  cmd2.ExecuteNonQuery();  using var cmd = new NpgsqlCommand("DELETE from \"Characters\" WHERE \"id\" = " + c\_id, db);  cmd.ExecuteNonQuery();  }  public void character\_edit(string c\_name, int c\_HP, int c\_lvl, int c\_id)  {  using var cmd = new NpgsqlCommand("UPDATE \"Characters\" SET \"Character\_name\" = @c\_name, \"HP\"= @c\_HP, \"Level\"= @c\_lvl WHERE \"id\" = " + c\_id, db);  cmd.Parameters.AddWithValue("@c\_name", c\_name);  cmd.Parameters.AddWithValue("@c\_HP", c\_HP);  cmd.Parameters.AddWithValue("@c\_lvl", c\_lvl);  cmd.ExecuteNonQuery();  }  public void character\_generation(int num)  {  using var cmd = new NpgsqlCommand("INSERT INTO \"Characters\" (\"Character\_name\", \"HP\", \"Level\") SELECT chr(trunc(65 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) , trunc(random() \* 500 + 20), trunc(random() \* 500 + 20) FROM generate\_series(1, @num)", db);  cmd.Parameters.AddWithValue("@num", num);  cmd.ExecuteNonQuery();  }  #endregion    #region Items  public string items\_print()  {  using var cmd = new NpgsqlCommand("SELECT \* from \"Items\" ORDER BY id ", db);  using NpgsqlDataReader rdr = cmd.ExecuteReader();  string items = "";  while (rdr.Read())  {  items += ". id: ";  items += rdr.GetInt32(0);  if (items.Length == 0)  {  break;  }  items += ". Name: ";  items += rdr.GetString(1);  items += " ATK: ";  items += rdr.GetInt32(2);  items += "\n";  }  return items;  }  public string item\_get\_by\_id(int dir\_id)  {  using var cmd = new NpgsqlCommand("SELECT \* FROM \"Items\" WHERE \"id\"=" + dir\_id, db);  using NpgsqlDataReader rdr = cmd.ExecuteReader();  string items = "";  while (rdr.Read())  {  items += ". id: ";  items += rdr.GetInt32(0);  if (items.Length == 0)  {  break;  }  items += ". Name: ";  items += rdr.GetString(1);  items += " ATK: ";  items += rdr.GetInt32(2);  items += "\n";  }  return items;  }  public int item\_add(string i\_name, int i\_ATK)  {  using var cmd = new NpgsqlCommand("INSERT INTO \"Items\"(\"name\", \"ATK\") VALUES(@name, @ATK)", db);  cmd.Parameters.AddWithValue("name", i\_name);  cmd.Parameters.AddWithValue("ATK", i\_ATK);  cmd.Prepare();  cmd.ExecuteNonQuery();  using var cmd2 = new NpgsqlCommand("SELECT \"id\" FROM \"Items\" WHERE id = (SELECT MAX(id) from \"Items\")", db);  using NpgsqlDataReader rdr = cmd2.ExecuteReader();  int new\_id = 0;  while (rdr.Read())  {  new\_id = rdr.GetInt32(0);  }  return new\_id;  }  public void item\_delete(int i\_id)  {  using var cmd2 = new NpgsqlCommand("DELETE from \"Items\" WHERE \"id\"= " + i\_id, db);  cmd2.ExecuteNonQuery();  using var cmd = new NpgsqlCommand("DELETE from \"Items\" WHERE \"id\" = " + i\_id, db);  cmd.ExecuteNonQuery();  }  public void item\_edit(string i\_name, int i\_ATK, int item\_id)  {  using var cmd = new NpgsqlCommand("UPDATE \"Items\" SET \"name\" = @i\_name, \"ATK\" = @i\_ATK WHERE \"id\" = " + item\_id, db);  cmd.Parameters.AddWithValue("@i\_name", i\_name);  cmd.Parameters.AddWithValue("@i\_ATK", i\_ATK);  cmd.ExecuteNonQuery();  }  public void Item\_generation(int num)  {  using var cmd = new NpgsqlCommand("INSERT INTO \"Items\" (\"name\", \"ATK\") SELECT chr(trunc(65 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int), trunc(random() \* 500 + 20) FROM generate\_series(1, @num)", db);  cmd.Parameters.AddWithValue("@num", num);  cmd.ExecuteNonQuery();  }  #endregion  #region accounts  public string account\_print()  {  using var cmd = new NpgsqlCommand("SELECT \* from \"accounts\" ORDER BY id ", db);  using NpgsqlDataReader rdr = cmd.ExecuteReader();  string accounts = "";  while (rdr.Read())  {  accounts += rdr.GetInt32(0);  if (accounts.Length == 0)  {  break;  }  accounts += ". Name: ";  accounts += rdr.GetString(1);  accounts += " Password: ";  accounts += rdr.GetString(2);  accounts += "\n";  }  return accounts;  }  public string account\_get\_by\_id(int acc\_id)  {  using var cmd = new NpgsqlCommand("SELECT \* FROM \"accounts\" WHERE \"id\"=" + acc\_id, db);  using NpgsqlDataReader rdr = cmd.ExecuteReader();  string acc = "";  while (rdr.Read())  {  acc += rdr.GetInt32(0);  if (acc.Length == 0)  {  break;  }  acc += ". Name: ";  acc += rdr.GetString(1);  acc += " Password: ";  acc += rdr.GetString(2);  acc += "\n";  }  return acc;  }  public int account\_add(string name, string pass)  {  using var cmd = new NpgsqlCommand("INSERT INTO \"accounts\"(\"name\", \"pword\") VALUES(@name, @pass)", db);  cmd.Parameters.AddWithValue("name", name);  cmd.Parameters.AddWithValue("pass", pass);  cmd.Prepare();  cmd.ExecuteNonQuery();  using var cmd2 = new NpgsqlCommand("SELECT id FROM \"accounts\" WHERE id = (SELECT MAX(id) from \"accounts\")", db);  using NpgsqlDataReader rdr = cmd2.ExecuteReader();  int new\_id = 0;  while (rdr.Read())  {  new\_id = rdr.GetInt32(0);  }  return new\_id;  }  public void account\_delete(int acc\_id)  {  using var cmd2 = new NpgsqlCommand("DELETE from \"accounts\" WHERE \"id\" = " + acc\_id, db);  cmd2.ExecuteNonQuery();  using var cmd = new NpgsqlCommand("DELETE from \"accounts\" WHERE \"id\" = " + acc\_id, db);  cmd.ExecuteNonQuery();  }  public void account\_edit(string a\_name, string a\_pass, int acc\_id)  {  using var cmd = new NpgsqlCommand("UPDATE \"accounts\" SET \"name\" = @a\_name, pword = @a\_pass WHERE \"id\" = " + acc\_id, db);  cmd.Parameters.AddWithValue("@a\_name", a\_name);  cmd.Parameters.AddWithValue("@a\_pass", a\_pass);  cmd.ExecuteNonQuery();  }  public void acc\_generation(int num)  {  using var cmd = new NpgsqlCommand("INSERT INTO \"accounts\" (\"name\", \"pword\") SELECT chr(trunc(65 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int), chr(trunc(65 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) || chr(trunc(97 + random()\*25)::int) FROM generate\_series(1, @num)", db);  cmd.Parameters.AddWithValue("@num", num);  cmd.ExecuteNonQuery();  }  #endregion  #region Characters\_items  public string acc\_item\_print()  {  using var cmd = new NpgsqlCommand("SELECT \"Characters\".\"id\" AS \"char\_id\", \"Characters\".\"Character\_name\", \"Characters-Items\".\"item\_id\", \"Items\".\"name\" from \"Characters\" join \"Characters-Items\" on (\"Characters\".\"id\" = \"Characters-Items\".\"char\_id\") join \"Items\" on (\"Items\".\"id\"=\"Characters-Items\".\"item\_id\")", db);  using NpgsqlDataReader rdr = cmd.ExecuteReader();  string Char\_item = "";  while (rdr.Read())  {  Char\_item += rdr.GetInt32(0);  if (Char\_item.Length == 0)  {  break;  }  Char\_item += ". Character Name: ";  Char\_item += rdr.GetString(1);  Char\_item += " <---> ";  Char\_item += rdr.GetInt32(2);  Char\_item += ". Item name: ";  Char\_item += rdr.GetString(3);  Char\_item += "\n";  }  return Char\_item;  }  public void character\_item\_add(int c\_id, int i\_id)  {  using var cmd = new NpgsqlCommand("INSERT INTO \"Characters-Items\"(\"char\_id\", \"item\_id\") VALUES((SELECT \"id\" from \"Characters\" where \"id\" = @c\_id), (SELECT \"id\" from \"Items\" where \"id\" = @i\_id))", db);  cmd.Parameters.AddWithValue("c\_id", c\_id);  cmd.Parameters.AddWithValue("i\_id", i\_id);  cmd.Prepare();  cmd.ExecuteNonQuery();  }  public string character\_item\_delete(int l\_id)  {  using var cmd = new NpgsqlCommand("DELETE from \"Characters-Items\" WHERE \"link\_id\"= @l\_id ", db);  cmd.Parameters.AddWithValue("l\_id", l\_id);  return cmd.ExecuteNonQuery().ToString();  }  public void acc\_item\_generation(int num)  {  using var cmd = new NpgsqlCommand("INSERT INTO \"Characters-Items\" (\"char\_id\", \"item\_id\") SELECT gen\_char\_id(), gen\_item\_id() FROM generate\_series(1, @num)", db);  cmd.Parameters.AddWithValue("@num", num);  cmd.ExecuteNonQuery();  }  #endregion  #region search  public string search\_option\_1(int s\_lvl, int e\_lvl, int s\_id, int e\_id, int s\_ATK, int e\_ATK)  {  using var cmd = new NpgsqlCommand("SELECT \"Characters\".\"id\" AS \"Characters-Items.char\_id\", \"Characters\".\"Character\_name\", \"Characters\".\"Level\", \"Characters\".\"HP\", \"Characters-Items\".\"item\_id\", \"Items\".\"name\", \"Items\".\"ATK\" from \"Characters\" join \"Characters-Items\" on (\"Characters\".\"id\" = \"Characters-Items\".\"char\_id\")join \"Items\" on (\"Characters-Items\".\"item\_id\" = \"Items\".\"id\") WHERE \"Characters\".\"Level\" BETWEEN @s\_lvl AND @e\_lvl AND \"Characters\".\"id\" BETWEEN @s\_id AND @e\_id AND \"Items\".\"ATK\" BETWEEN @s\_ATK AND @e\_ATK", db);  cmd.Parameters.AddWithValue("s\_lvl", s\_lvl);  cmd.Parameters.AddWithValue("e\_lvl", e\_lvl);  cmd.Parameters.AddWithValue("s\_id", s\_id);  cmd.Parameters.AddWithValue("e\_id", e\_id);  cmd.Parameters.AddWithValue("s\_ATK", s\_ATK);  cmd.Parameters.AddWithValue("e\_ATK", e\_ATK);  TimeSpan ts = DateTime.Now.TimeOfDay;  var sw = new Stopwatch();  sw.Start();  using NpgsqlDataReader rdr = cmd.ExecuteReader();  string search = "";  while (rdr.Read())  {  search += rdr.GetInt32(0);  if (search.Length == 0)  {  break;  }  search += ". Character Name: ";  search += rdr.GetString(1);  search += " Level:";  search += rdr.GetInt32(2);  search += " HP: ";  search += rdr.GetInt32(3);  search += " <---> ";  search += rdr.GetInt32(4);  search += ". Item Name: ";  search += rdr.GetString(5);  search += " ATK ";  search += rdr.GetInt32(6);  search += "\n";  }  var elapsed = sw.ElapsedMilliseconds;  Console.WriteLine($"Query Executed and Results Returned in 0.{elapsed.ToString()}sec");  return search;  }  public string search\_option\_2(string c\_name, int e\_lvl, int s\_lvl, string i\_name)  {  using var cmd = new NpgsqlCommand("SELECT \"Characters\".\"id\" AS \"Characters-Items.char\_id\", \"Characters\".\"Character\_name\", \"Characters\".\"Level\", \"Characters\".\"HP\", \"Characters-Items\".\"item\_id\", \"Items\".\"name\", \"Items\".\"ATK\" from \"Characters\" join \"Characters-Items\" on (\"Characters\".\"id\" = \"Characters-Items\".\"char\_id\")join \"Items\" on (\"Characters-Items\".\"item\_id\" = \"Items\".\"id\") WHERE \"Characters\".\"Level\" BETWEEN @e\_lvl AND @s\_lvl AND \"Items\".\"name\" like '%" + i\_name + "%' AND \"Characters\".\"Character\_name\" like '%"+c\_name+"%'", db);  cmd.Parameters.AddWithValue("i\_name", i\_name);  cmd.Parameters.AddWithValue("c\_name", c\_name);  cmd.Parameters.AddWithValue("e\_lvl", e\_lvl);  cmd.Parameters.AddWithValue("s\_lvl", s\_lvl);  var sw = new Stopwatch();  sw.Start();  using NpgsqlDataReader rdr = cmd.ExecuteReader();  string search = "";  while (rdr.Read())  {  search += rdr.GetInt32(0);  if (search.Length == 0)  {  break;  }  search += ". Character Name: ";  search += rdr.GetString(1);  search += ". Level: ";  search += rdr.GetInt32(2);  search += ". HP: ";  search += rdr.GetInt32(3);  search += " ---> ";  search += rdr.GetInt32(4);  search += ". Item name: ";  search += rdr.GetString(5);  search += " ATK:";  search += rdr.GetInt32(6);  search += "\n";  }  var elapsed = sw.ElapsedMilliseconds;  Console.WriteLine($"Query Executed and Results Returned in 0.{elapsed.ToString()}sec");  return search;  }  public string search\_option\_3(string c\_name, int e\_hp, int s\_hp, string i\_name)  {  using var cmd = new NpgsqlCommand("SELECT \"Characters\".\"id\" AS \"Characters-Items.char\_id\", \"Characters\".\"Character\_name\", \"Characters\".\"Level\", \"Characters\".\"HP\", \"Characters-Items\".\"item\_id\", \"Items\".\"name\", \"Items\".\"ATK\" from \"Characters\" join \"Characters-Items\" on (\"Characters\".\"id\" = \"Characters-Items\".\"char\_id\")join \"Items\" on (\"Characters-Items\".\"item\_id\" = \"Items\".\"id\") WHERE \"Characters\".\"Level\" BETWEEN @e\_hp AND @s\_hp AND \"Items\".\"name\" like '%" + i\_name + "%' AND \"Characters\".\"Character\_name\" like '%" + c\_name + "%'", db);  cmd.Parameters.AddWithValue("i\_name", i\_name);  cmd.Parameters.AddWithValue("c\_name", c\_name);  cmd.Parameters.AddWithValue("e\_hp", e\_hp);  cmd.Parameters.AddWithValue("s\_hp", s\_hp);  var sw = new Stopwatch();  sw.Start();  using NpgsqlDataReader rdr = cmd.ExecuteReader();  string search = "";  while (rdr.Read())  {  search += rdr.GetInt32(0);  if (search.Length == 0)  {  break;  }  search += ". Character Name: ";  search += rdr.GetString(1);  search += ". Level: ";  search += rdr.GetInt32(2);  search += ". HP: ";  search += rdr.GetInt32(3);  search += " ---> ";  search += rdr.GetInt32(4);  search += ". Item name: ";  search += rdr.GetString(5);  search += " ATK:";  search += rdr.GetInt32(6);  search += "\n";  }  var elapsed = sw.ElapsedMilliseconds;  Console.WriteLine($"Query Executed and Results Returned in 0.{elapsed.ToString()}sec");  return search;  }  #endregion  }  } |
| ***View.cs***  using System;  using lab2.MVC;  namespace lab2.MVC  {  class View  {  public string entity()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Menu:");  Console.ForegroundColor = ConsoleColor.Gray;  Console.WriteLine("1.Characters\n2.Items\n3.Accounts\n4.Characters - Items\n5.Search operations\n6.Exit");  return Console.ReadLine();  }  #region Character  public string Character()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Character Menu:");  Console.ForegroundColor = ConsoleColor.Gray;  Console.WriteLine("1.Print list of Characters\n2.Print character by ID\n3.Add chracter\n4.Delete character by ID\n5.Edit character by ID\n6.Random Generation of chracters\n7.Choose another entity");  return Console.ReadLine();  }  public string chracter\_get\_name()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Chracter name:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string char\_get\_level()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Level:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string char\_get\_hp()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Character HP:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string char\_get\_ATK()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Character ATK:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  #endregion  #region Items  public string item()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Item Menu:");  Console.ForegroundColor = ConsoleColor.Gray;  Console.WriteLine("1.Print list of items\n2.Print item by ID\n3.Add item\n4.Delete item by ID\n5.Edit item by ID\n6.Random Generation of items\n7.Choose another entity");  return Console.ReadLine();  }  public string Item\_get\_name()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Item name:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  #endregion  #region Account  public string account()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Account Menu:");  Console.ForegroundColor = ConsoleColor.Gray;  Console.WriteLine("1.Print list of Accounts\n2.Print account by ID\n3.Add ccount\n4.Delete account by ID\n5.Edit account by ID\n6.Random Generation of accounts\n7.Choose another entity");  string a = Console.ReadLine();  return a;  }  public string acc\_get\_name()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Account Name:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string acc\_get\_pass()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Account password:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  #endregion  #region Characters-Items  public string char\_it()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Characters-items Menu:");  Console.ForegroundColor = ConsoleColor.Gray;  Console.WriteLine("1.Print connections\n2.Add connection\n3.Delete connection\n4.Random Generation of connections\n5.Choose another entity");  string a = Console.ReadLine();  return a;  }  public string get\_char\_id()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input ID of a Character:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string get\_item\_id()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input ID of an Item:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  #endregion  #region search  public string search()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Search Operations:");  Console.ForegroundColor = ConsoleColor.Gray;  Console.WriteLine("1.Search for the character with limited id,ATK and Level\n2.Search for the character with limited lvl and similia char/weapon name\n3.Search for the character with limited HP and similiar char/weapon name\n4.Go to entities menu");  string a = Console.ReadLine();  return a;  }  public string search\_s\_lvl()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input lower border of the lvl:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string search\_e\_lvl()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input upper border of the lvl:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string search\_s\_id()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input lower border of the ID interval:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string search\_e\_id()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input uper border of the ID interval:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string search\_s\_ATK()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input lower border of the ATK interval:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string search\_e\_ATK()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input upper border of the ATK interval:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string search\_c\_name()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input substring from characters`s name:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string search\_i\_name()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input substring from item`s name:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string search\_s\_hp()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input lower border of the HP interval:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string search\_e\_hp()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input upper border of the HP interval:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  #endregion  public void print(string entities)  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("A list of entities:");  Console.ForegroundColor = ConsoleColor.Gray;  Console.WriteLine(entities);  }  public string get\_id()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input ID of entity:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  public string get\_num()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Input number of randomly generated entities:");  Console.ForegroundColor = ConsoleColor.Gray;  return Console.ReadLine();  }  #region errors  public void err\_wrong\_entity()  {  Console.ForegroundColor = ConsoleColor.Red;  Console.WriteLine($"The entity with such a number does not exist or you've entered a string");  Console.ForegroundColor = ConsoleColor.Gray;  }  public void err\_wrong\_option()  {  Console.ForegroundColor = ConsoleColor.Red;  Console.WriteLine($"The option with such a number does not exist or you've entered a string");  Console.ForegroundColor = ConsoleColor.Gray;  }  public void err\_empty\_table(string entity)  {  Console.ForegroundColor = ConsoleColor.Red;  Console.WriteLine(entity + " table is empty");  Console.ForegroundColor = ConsoleColor.Gray;  }  public void err\_wrong\_ID(string entity)  {  Console.ForegroundColor = ConsoleColor.Red;  Console.WriteLine(entity + " with ID does not exist or you've entered a string");  Console.ForegroundColor = ConsoleColor.Gray;  }  public void err\_empty(string entity)  {  Console.ForegroundColor = ConsoleColor.Red;  Console.WriteLine(entity + " cannot be empty");  Console.ForegroundColor = ConsoleColor.Gray;  }  public void err\_number(string entity)  {  Console.ForegroundColor = ConsoleColor.Red;  Console.WriteLine(entity + " shold be a number");  Console.ForegroundColor = ConsoleColor.Gray;  }  public void err\_generation()  {  Console.ForegroundColor = ConsoleColor.Red;  Console.WriteLine("Number shold be between 0 and 100 000");  Console.ForegroundColor = ConsoleColor.Gray;  }  public void err\_connection()  {  Console.ForegroundColor = ConsoleColor.Red;  Console.WriteLine("Connection does not exist");  Console.ForegroundColor = ConsoleColor.Gray;  }  #endregion  #region successfull  public void successfull\_operation(string entity, int ID, string operation)  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine(entity + " with ID " + ID + " " + operation + " successfully");  Console.ForegroundColor = ConsoleColor.Gray;  }  public void successfull\_generation(string entity, int num)  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine(num + " " + entity + " generated successfully");  Console.ForegroundColor = ConsoleColor.Gray;  }  public void successfull\_connection()  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("New connection added successfully");  Console.ForegroundColor = ConsoleColor.Gray;  }  public void successfull\_connection\_delete(int link)  {  Console.ForegroundColor = ConsoleColor.Cyan;  Console.WriteLine("Connection " + link + " deleted successfully");  Console.ForegroundColor = ConsoleColor.Gray;  }  #endregion  }  } |
| ***Controller.cs***  using System;  using lab2.MVC;  namespace lab2.MVC  {  class Controller  {  Model model = new Model();  View view = new View();  public int entity\_menu()  {  while (1 == 1)  {  int entity = 0;  Int32.TryParse(view.entity(), out entity);  if (entity == 1)  {  character\_menu();  break;  }  else if (entity == 2)  {  item\_menu();  break;  }  else if (entity == 3)  {  account\_menu();  break;  }  else if (entity == 4)  {  Character\_Items\_menu();  break;  }  else if (entity == 5)  {  search\_menu();  break;  }  else if (entity == 6)  {  return 1;  }  else  {  view.err\_wrong\_entity();  }  }  return 0;  }  #region Character  private void character\_menu()  {  while (1 == 1)  {  int character = 0;  Int32.TryParse(view.Character(), out character);  if (character == 1)  {  string characters = model.character\_print();  if (characters.Length == 0)  {  view.err\_empty\_table("Characters");  }  else  {  view.print(characters);  }  }  else if (character == 2)  {  int id = 0;  Int32.TryParse(view.get\_id(), out id);  string characters = model.character\_get\_by\_id(id);  if (characters.Length == 0)  {  view.err\_wrong\_ID("Character ");  }  else  {  view.print(characters);  }  }  else if (character == 3)  {  string name = view.chracter\_get\_name();  while (name.Length == 0)  {  view.err\_empty("Character name");  name = view.chracter\_get\_name();  }  int Level = Convert.ToInt32(view.char\_get\_level());  int hp = Convert.ToInt32(view.char\_get\_hp());  int new\_id = model.character\_add(name, Level, hp);  view.successfull\_operation("Character", new\_id, "added");  }  else if (character == 4)  {  int id = 0;  Int32.TryParse(view.get\_id(), out id);  if (model.character\_get\_by\_id(id).Length == 0)  {  view.err\_wrong\_ID("Character ");  }  else  {  model.character\_delete(id);  view.successfull\_operation("Character", id, "deleted");  }  }  else if (character == 5)  {  int id = 0;  Int32.TryParse(view.get\_id(), out id);  if (model.character\_get\_by\_id(id).Length == 0)  {  view.err\_wrong\_ID("Character ");  }  else  {  string name = view.chracter\_get\_name();  while (name.Length == 0)  {  view.err\_empty("Character name");  name = view.chracter\_get\_name();  }  int Level = Convert.ToInt32(view.char\_get\_level());  int hp = Convert.ToInt32(view.char\_get\_hp());  model.character\_edit(name, Level, hp, id);  view.successfull\_operation("Character", id, "edited");  }  }  else if (character == 6)  {  int num = 0;  while (!Int32.TryParse(view.get\_num(), out num) || num <= 0 || num > 100000)  {  view.err\_generation();  }  model.character\_generation(num);  view.successfull\_generation("Character", num);  }  else if (character == 7)  {  break;  }  else  {  view.err\_wrong\_option();  }  }  }  #endregion  #region Item  private void item\_menu()  {  while (1 == 1)  {  int item = 0;  Int32.TryParse(view.item(), out item);  if (item == 1)  {  string directors = model.items\_print();  if (directors.Length == 0)  {  view.err\_empty\_table("Item");  }  else  {  view.print(directors);  }  }  else if (item == 2)  {  int id = 0;  Int32.TryParse(view.get\_id(), out id);  string directors = model.item\_get\_by\_id(id);  if (directors.Length == 0)  {  view.err\_wrong\_ID("Item");  }  else  {  view.print(directors);  }  }  else if (item == 3)  {  string name = view.Item\_get\_name();  while (name.Length == 0)  {  view.err\_empty("Item name");  name = view.Item\_get\_name();  }  int ATK = Convert.ToInt32(view.char\_get\_ATK());  int new\_id = model.item\_add(name, ATK);  view.successfull\_operation("Item", new\_id, "added");  }  else if (item == 4)  {  int id = 0;  Int32.TryParse(view.get\_id(), out id);  if (model.item\_get\_by\_id(id).Length == 0)  {  view.err\_wrong\_ID("Item");  }  else  {  model.item\_delete(id);  view.successfull\_operation("Item", id, "deleted");  }  }  else if (item == 5)  {  int id = 0;  Int32.TryParse(view.get\_id(), out id);  if (model.item\_get\_by\_id(id).Length == 0)  {  view.err\_wrong\_ID("Item");  }  else  {  string name = view.Item\_get\_name();  while (name.Length == 0)  {  view.err\_empty("Item name");  name = view.Item\_get\_name();  }  int ATK = Convert.ToInt32(view.char\_get\_ATK());  model.item\_edit(name, ATK, id);  view.successfull\_operation("Item", id, "edited");  }  }  else if (item == 6)  {  int num = 0;  while (!Int32.TryParse(view.get\_num(), out num) || num <= 0 || num > 100000)  {  view.err\_generation();  }  model.Item\_generation(num);  view.successfull\_generation("items", num);  }  else if (item == 7)  {  break;  }  else  {  view.err\_wrong\_option();  }  }  }  #endregion  #region account  private void account\_menu()  {  while (1 == 1)  {  int acc = 0;  Int32.TryParse(view.account(), out acc);  if (acc == 1)  {  string accounts = model.account\_print();  if (accounts.Length == 0)  {  view.err\_empty\_table("Accounts");  }  else  {  view.print(accounts);  }  }  else if (acc == 2)  {  int id = 0;  Int32.TryParse(view.get\_id(), out id);  string accounts = model.account\_get\_by\_id(id);  if (accounts.Length == 0)  {  view.err\_wrong\_ID("Account");  }  else  {  view.print(accounts);  }  }  else if (acc == 3)  {  string name = view.acc\_get\_name();  while (name.Length == 0)  {  view.err\_empty("Award category");  name = view.acc\_get\_name();  }  string pass = view.acc\_get\_pass();  while (pass.Length==0)  {  view.err\_number("Account password");  pass = view.acc\_get\_pass();  }  int new\_id = model.account\_add(name, pass);  view.successfull\_operation("Account", new\_id, "added");  }  else if (acc == 4)  {  int id = 0;  Int32.TryParse(view.get\_id(), out id);  if (model.account\_get\_by\_id(id).Length == 0)  {  view.err\_wrong\_ID("Account");  }  else  {  model.account\_delete(id);  view.successfull\_operation("Account", id, "deleted");  }  }  else if (acc == 5)  {  int id = 0;  Int32.TryParse(view.get\_id(), out id);  if (model.item\_get\_by\_id(id).Length == 0)  {  view.err\_wrong\_ID("Account");  }  else  {  string name = view.acc\_get\_name();  while (name.Length == 0)  {  view.err\_empty("Account name");  name = view.acc\_get\_name();  }  string pass = view.acc\_get\_pass();  while (pass.Length==0)  {  view.err\_number("Account password");  pass = view.acc\_get\_pass();  }  model.account\_edit(name, pass, id);  view.successfull\_operation("Account", id, "edited");  }  }  else if (acc == 6)  {  int num = 0;  while (!Int32.TryParse(view.get\_num(), out num) || num <= 0 || num > 100000)  {  view.err\_generation();  }  model.acc\_generation(num);  view.successfull\_generation("Accounts", num);  }  else if (acc == 7)  {  break;  }  else  {  view.err\_wrong\_option();  }  }  }  #endregion  #region Characters-Items  private void Character\_Items\_menu()  {  while (1 == 1)  {  int ci = 0;  Int32.TryParse(view.char\_it(), out ci);  if (ci == 1)  {  string characters\_items = model.acc\_item\_print();  if (characters\_items.Length == 0)  {  view.err\_empty\_table("Characters - Items");  }  else  {  view.print(characters\_items);  }  }  else if (ci == 2)  {  int char\_id = 0;  Int32.TryParse(view.get\_char\_id(), out char\_id);  while (model.character\_get\_by\_id(char\_id).Length == 0)  {  view.err\_wrong\_ID("Character");  Int32.TryParse(view.get\_char\_id(), out char\_id);  }  int item\_id = 0;  Int32.TryParse(view.get\_item\_id(), out item\_id);  while (model.item\_get\_by\_id(item\_id).Length == 0)  {  view.err\_wrong\_ID("Item");  Int32.TryParse(view.get\_item\_id(), out item\_id);  }  model.character\_item\_add(char\_id, item\_id);  view.successfull\_connection();  }  else if (ci == 3)  {  int link = 0;  Int32.TryParse(view.get\_item\_id(), out link);  string del = model.character\_item\_delete(link);  if (del == "0")  {  view.err\_connection();  }  else  {  view.successfull\_connection\_delete(link);  }  }  else if (ci == 4)  {  int num = 0;  while (!Int32.TryParse(view.get\_num(), out num) || num <= 0 || num > 100000)  {  view.err\_generation();  }  model.acc\_item\_generation(num);  view.successfull\_generation("Connections", num);  }  else if (ci == 5)  {  break;  }  else  {  view.err\_wrong\_option();  }  }  }  #endregion  #region search  private void search\_menu()  {  while (1 == 1)  {  int search = 0;  Int32.TryParse(view.search(), out search);  if (search == 1)  {  int s\_lvl = 0;  while (!Int32.TryParse(view.search\_s\_lvl(), out s\_lvl))  {  view.err\_number("Input");  }  int e\_lvl = 0;  while (!Int32.TryParse(view.search\_e\_lvl(), out e\_lvl))  {  view.err\_number("Input");  }  int s\_id = 0;  while (!Int32.TryParse(view.search\_s\_id(), out s\_id))  {  view.err\_number("Input");  }  int e\_id = 0;  while (!Int32.TryParse(view.search\_e\_id(), out e\_id))  {  view.err\_number("Input");  }  int s\_ATK = 0;  while (!Int32.TryParse(view.search\_s\_ATK(), out s\_ATK))  {  view.err\_number("Input");  }  int e\_ATK = 0;  while (!Int32.TryParse(view.search\_e\_ATK(), out e\_ATK))  {  view.err\_number("Input");  }  string searches = model.search\_option\_1(s\_lvl, e\_lvl, s\_id, e\_id, s\_ATK, e\_ATK);  if (searches.Length == 0)  {  view.err\_empty\_table("This");  }  else  {  view.print(searches);  }  }  else if (search == 2)  {  int s\_lvl = 0;  while (!Int32.TryParse(view.search\_s\_lvl(), out s\_lvl))  {  view.err\_number("Input");  }  int e\_lvl = 0;  while (!Int32.TryParse(view.search\_e\_lvl(), out e\_lvl))  {  view.err\_number("Input");  }  string c\_name = view.search\_c\_name();  while (c\_name.Length == 0)  {  view.err\_empty("Substring");  c\_name = view.search\_c\_name();  }  string i\_name = view.search\_i\_name();  while (i\_name.Length == 0)  {  view.err\_empty("Substring");  i\_name = view.search\_i\_name();  }  string searches = model.search\_option\_2(c\_name, e\_lvl, s\_lvl, i\_name);  if (searches.Length == 0)  {  view.err\_empty\_table("This");  }  else  {  view.print(searches);  }  }  else if (search == 3)  {  string c\_name = view.search\_c\_name();  while (c\_name.Length == 0)  {  view.err\_empty("Substring");  c\_name = view.search\_c\_name();  }  string i\_name = view.search\_i\_name();  while (i\_name.Length == 0)  {  view.err\_empty("Substring");  i\_name = view.search\_i\_name();  }  int s\_hp = 0;  while (!Int32.TryParse(view.search\_s\_hp(), out s\_hp))  {  view.err\_number("Input");  }  int e\_hp = 0;  while (!Int32.TryParse(view.search\_e\_hp(), out e\_hp))  {  view.err\_number("Input");  }  string searches = model.search\_option\_3(c\_name, e\_hp, s\_hp, i\_name);  if (searches.Length == 0)  {  view.err\_empty\_table("This");  }  else  {  view.print(searches);  }  }  else if (search == 4)  {  break;  }  else  {  view.err\_wrong\_option();  }  }  }  #endregion  }  } |