Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Department of Applied
Radio Electronics

[RE-88] .NET TECHNOLOGIES FOR SOFTWARE DEVELOPMENT

Curriculum of the academic discipline (Syllabus)

Course details

Level of higher education	First (bachelor's)
Field of knowledge	G - Engineering, manufacturing, and construction
Special	G5 - Electronics, electronic communications, instrument engineering, and radio engineering
Educational program	All educational programs
Subject status	Elective (F-catalog)
Form of higher education	Full-time
Year of training, semester	Available for selection starting from the 2nd year, fall semester
Scope of the discipline	4 credits (Lectures 16 hours, Practical 30 hours, Lab 0 hours, Independent work 74 hours)
Semester Control/control measures	Credit
Class schedule	https://schedule.kpi.ua
Language of instruction	Ukrainian
Information about the course leader/teachers	Lecturer: Nikitchuk A. V., Practical: Nikitchuk A. V.,

Course location https://do.ipo.kpi.ua/course/view.php?id=5927

Curriculum

1. Description of the course, its purpose, subject matter, and learning outcomes

Objective: to familiarize students with the basic concepts and features of the *Microsoft .NET (.NET)* platform, to acquire the knowledge and skills necessary for the industrial development of software products with the appropriate functionality for radio engineering information systems.

.NET is a platform that reflects the latest trends in development and offers many opportunities for novice specialists. Knowledge of the basics *of .NET* will allow you to be flexible in choosing specialization and programming field.

The main objectives of the discipline are:

1. Learning the basic concepts and components of the *.NET* platform.
2. Familiarization with *Visual Studio* development tools and environment to increase productivity in application development.
3. Mastering the C# programming language for creating desktop, web, and mobile applications on the *.NET* platform.
4. Using various libraries and frameworks that are part of the *.NET* ecosystem to implement various functionalities.
5. Introduction to web application development using *ASP.NET Core* technology, including working with *MVC* (*Model-View-Controller*).
6. Creating databases on the *MSSQL* server, connecting to the project, and using them.
7. Familiarization with the development of mobile and cross-platform applications on the *.NET* platform using *MAUI* (formerly *Xamarin*).
8. Learning programming principles and patterns, the basics of refactoring, for continuous improvement of software product quality.

As a result of studying the discipline, the following competencies are formed:

PC 19	Ability to apply object-oriented programming technology and basic design patterns when creating software with appropriate functionality for radio-technical information systems and implement programs in various programming environments.
PC 20	Ability to select methods and means of information processing using intelligent technologies.
PLO 4	Apply databases, mathematical and software for data processing and computer modeling of telecommunications and radio engineering systems, and intelligent radio electronics technologies.

Students will learn:
– the C# programming language;
– the basic concepts of *.NET* technology and the basic principles of creating software using it;
– the main libraries that make up *.NET*.

Students will be able to:
– work in the *Microsoft Visual Studio* programming environment;
– create programs in C# using object-oriented technology;
– create software with a graphical user interface (for desktop, web, and mobile use);
– develop software for interacting with databases;
– develop software for working in the Internet telecommunications network using *ASP.NET Core* technology.
In addition, .NET on C# can be used to create IoT (Internet of Things) applications for Raspberry Pi, HummingBoard, BeagleBoard, Pine A64, and others. Using existing open source libraries and frameworks to

interact with specialized hardware such as sensors, analog-to-digital converters, and LCD devices. Also, with .NET, you can create computer games for the Unity platform (this is how such well-known games as Hearthstone: Heroes of Warcraft, Cities Skylines, Cuphead, Pillars of Eternity, Pokemon Go, Subnautica, and others were created).

2. Prerequisites and postrequisites of the discipline (place in the structural-logical scheme of the relevant educational program)

Before starting to study the discipline, students should have:

- knowledge of computer science and basic programming skills (courses: Computer Science, Introduction to the Specialty);
- English language skills (or skills in using online translators).

Related disciplines:

- Software Quality Control.
- Programming of Embedded Systems of the Internet of Things.
- Computer networks and security using CISCO technologies.

3. Course content

Topic 1. Computers, programming, and Microsoft.NET
Topic 2. Basics of program execution and the C# programming language
Topic 3. Basic C# operators
Topic 4. Features of OOP implementation and interaction between classes
Topic 5. Exception handling. LINQ basics
Topic 6. Application programming interface
Topic 7. Creating web applications using the MVC architectural pattern
Topic 8. Software design principles and patterns

4. Learning materials and resources

Electronic resources

1. Microsoft library of documentation and learning resources for developers and other professionals working with technologies — https://docs.microsoft.com/uk-ua/
2. W3Schools — the largest free educational website for developers and online programming training. 3 billion pages are viewed annually. 60 million visitors per month — https://www.w3schools.com/
3. C# Corner — A global online community of software developers. In 2021, C# Corner served 29.4 million visitors — https://www.c-sharpcorner.com/
4. Programiz is an educational website for learning programming. Millions of users view tutorials and examples from around the world — https://www.programiz.com/
5. Stackify helps developers write better code. Their products allow you to test your code as you write it. This helps developers fix performance issues early on and better verify their code — https://stackify.com/

Books

1. Head First Design Patterns, Eric Freeman, Elizabeth Robson, Katie Sierra, Bert Bates (Ukrainian)
2. Clean Code, Robert Martin (Ukrainian)
3. C#: Learn C# in One Day and Learn It Well. C# for Beginners with Hands-on Project, LCF Publishing, 161 pages (English)
4. C# 8.0 and .NET Core 3.0, Mark J. Price (English)
5. C# in Depth: Programming Details, John Skeet (English)

Educational content

5. Methodology for mastering the academic discipline (educational component)

Class	Description
Topic 1. Computers, programming, and *Microsoft.NET*	
Lecture	Organizational aspects. Computer system. Software, computer programs. Operating system. Programming languages. How a computer processes programs. Microsoft.NET (.NET) ecosystem.
Topic 2. Basics of program execution and the C# programming language	
Lecture 2	Common Type System (CTS). Common Language Specification (CLS). Common Language Runtime (CLR). Variables and constants. Data types in .NET. Data categories: value types and reference types.
PR 1	Introduction to Visual Studio, code editor, solutions, projects, and their testing
Topic 3. Basic C# operators	
Lecture 3	Operators: arithmetic, assignment, logical, comparison, equality, Boolean. Program structure. Namespaces. Selection operators. Iteration operators. Arrays. OOP paradigm.
PR 2	Mathematical operations. Conditional and loop operators.
PR 3	One-dimensional arrays. Multidimensional arrays.
PR 4	Methods
PR 5	Classes
Topic 4. Features of OOP implementation and interaction between classes	
Lecture	Features of OOP implementation (constructor; *this*; object creation; static class; destructor). Interaction between classes (association; composition; aggregation; inheritance). Polymorphism (during compilation; during execution; method hiding).
PR 6	Inheritance (creating a base class and a descendant class, declaring their contents, working with methods of such classes)
PR 7	Association, composition, and aggregation (abstract class, descendant classes, abstract methods, method overriding)
Topic 5. Exception handling. *LINQ* basics	
Lecture 5	E x ceptions and exception handling. *LINQ* (*Language Integrated Query*). Queries to different data sources or data formats.
PR 8	Exceptions and handling of exceptional situations (checking arguments, using *try...catch* constructs, *throw* statements).
PR 9	Executing data queries using LINQ.
Topic 6. Application Programming Interface	
Lecture 6	Creating your first program with a graphical interface (form). Adding a control to a form. Creating event handlers. Adjusting size and scale. Examples of program creation.
PR 10	Creating a program with a graphical interface based on the *Windows Forms (.NET)* project.
PR 11	Creating a database and connecting it to the project. Debugging the program to work with the database. Publishing and testing the application.
Topic 7. Creating web applications using the MVC architectural pattern	
Lecture 7	*ASP.NET* is a technology for creating web applications and web services from Microsoft. The *Model-View-Controller* (*MVC*) architectural pattern divides the program into three main groups of components. Creating an *ASP.NET MVC* application, folder structure, adding components. Transferring data between the controller and the view.
Assignment 12	Creating a web application using *ASP.NET Core* technology and the *MVC* architectural pattern, part 1: user interface (*front-end*) and its connection to the controller (*back-end*).
PR 13	Connecting *Entity Framework Core*. Working with a database in an *ASP.NET Core MVC* application.
Topic 8. Software design principles and patterns	
Lecture 8	Principles of OOP programming and *SOLID* design. Programming patterns. Refactoring.
PR 14	Creating a cross-platform *MAUI (Multi-platform App UI)* application for mobile and desktop devices. Configuring the emulator and *Android* device.

PR 15	Cross-platform development. Migrating a project from *Windows Forms* to *MAUI*. Testing the application on an *Android/iOS* mobile device.

6. Independent student work

1. Throughout the semester:

- *Studying lecture material.*
- *Reviewing literary sources.*
- *Answering questions for self-assessment.*

2. During the week before the scheduled date:

- *Preparation for practical work.*
- *Prepare for writing the test.*
- *Preparation for completing the take-home test.*
- *Preparation for the exam.*

Policy and control

7. Academic discipline policy (educational component)

Rules for attending classes:

- *for lectures and practical classes - attendance at classes (Zoom video conferences) according to the schedule;*
- *independent study of the material using lecture recordings and other materials posted in the relevant distance learning course is permitted;*
- *asynchronous completion of practical assignments is permitted.*

Rules of conduct in class:

- *during classes, you must use the Internet to: complete assignments in the distance learning course; familiarize yourself with the links provided; access modern, organized sources of information;*
- *The use of mobile phones, laptops, and other devices is permitted.*

Rules for performing practical work:

- *if the teacher has questions about the results obtained, it is necessary to undergo an oral defense procedure (answer questions);*
- *The defense procedure is considered timely if it is completed during the class dedicated to the work or the next class according to the schedule.*

Rules for awarding bonus points:

- *Bonus points are awarded for completing additional tasks specified in the assignments.*

Rules for awarding penalty points:

- *Penalty points may be awarded for late submission/defense of practical work.*

Deadline and retake policy:

- *Tests, exams, and practical assignments must be completed by the last class of the semester.*

8. Types of assessment and the learning outcomes assessment rating system (LOAS)

- *Ongoing assessment: quizzes (tests) on lecture topics (8 points),* *completion of practical work (75 points), MCW (10 points), HCW (7 points).*

- *Calendar assessment: conducted twice per semester as monitoring of the current status of syllabus requirements fulfillment.*
- *Semester assessment: credit.*
- *Conditions for admission to semester assessment: semester rating above 60 points.*

Table of correspondence between rating points and grades on the university scale

Number of points	Grade
10	Excellent
94	Very good
84	Good
74-65	Satisfactory
64-60	Sufficient
Less than 60	Unsatisfactory
Admission requirements not met	Not admitted

9. Additional information on the discipline (educational component)

Description of material, technical, and informational support for the discipline

Classes are held on computers located in the PRE department's classroom. Students may also use their own computers. The main software used is *Microsoft Visual Studio Community*, a free, full-featured, extensible integrated development environment (*IDE*) for creating modern *Android*, *iOS*, and *Windows* applications, as well as web applications and cloud services.

The course syllabus (syllabus):
Compiled by Nikitchuk A. V.;
Approved by the PRE Department (Minutes No. 06/2025 dated 06/25/2025)
Approved by the methodological commission of the faculty/research institute (protocol No. 06/2025 dated 26.06.2025)