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

Department of Computer Engineering

User interface programming technologies (Front-end)

Working program of the academic discipline (Syllabus)

Details of the academic discipline

Level of higher education	***First (bachelor's)***
Branch of knowledge	*12 Information technologies*
Specialty	*121 Software Engineering* *123 Computer engineering*
Educational program	*Computer Systems Software Engineering* *Computer Systems and Networks*
Discipline status	*Selective*
Form of education	*intramural (full-time)/extramural*
Year of training, semester	*3rd year, autumn semester*
Scope of the discipline	*4 credits of 120 hours*
Semester control/ control measures	*Final test*
Lessons schedule	*//rozklad.kpi.ua*
Language of teaching	*English*
Information about the course leader / teachers	Lecturer: senior lecturer, Oleksii Aleshchenko alexey.aleshchenko@gmail.com Laboratory: senior lecturer, Oleksii Aleshchenko alexey.aleshchenko@gmail.com
Placement of the course	//comsys.kpi.ua

Program of educational discipline

1. Description of the educational discipline, its purpose, subject of study and learning outcomes

The discipline "Technologies of programming user interfaces (Front - end)" is aimed at students mastering the methods and means of developing user interfaces. Knowledge of the basics of programming user interfaces is necessary for creating software for computer systems, real-time systems, Internet applications, and mobile devices.

*The purpose of teaching the discipline "Technology of programming user interfaces (Front - end)" is: **ABILITY** : for abstract thinking, analysis and synthesis; work in a team; use modern methods and programming languages to develop algorithmic and software; create system and application software for computer systems and networks; to ensure the protection of information processed in computer and cyber-physical systems and networks for the purpose of implementing the established information security policy; systematically administer, use, adapt and operate existing information technologies and systems; identify, classify and describe the operation of software and technical means, computer and cyber-physical systems, networks and their components by using analytical methods and modeling methods; design, implement, administer and maintain global, local intelligent software-configured computer networks; develop, adapt, use software to improve the efficiency of high-performance computer systems;*

organization of computing processes in high-performance computer systems with different structural organization based on the use of the latest planning and dispatching technologies and modern operating systems;
KNOWLEDGE: *basics of project management; to identify, formulate and solve technical problems of the specialty, using the methods that are most suitable for achieving the set goals ; technical characteristics, design features, purpose and rules of operation of software and technical means of computer systems and networks for solving technical problems of the specialty;*
SKILLS *: conducting experiments, collecting data and modeling in computer systems; apply knowledge to identify, formulate and solve technical problems of the specialty, using methods that are most suitable for achieving the set goals; to solve problems of analysis and synthesis of means characteristic of the specialty; apply knowledge of technical characteristics, design features, purpose and rules of operation of software and technical means of computer systems and networks to solve technical problems of the specialty; develop software for embedded and distributed applications, mobile and hybrid systems, calculate, operate equipment typical for the specialty; perform experimental research on professional topics; perform parameter calculations of individual computer systems, computer networks; be able to create and maintain databases.*

2. requisites and post-requisites of the discipline (place in the structural and logical scheme of training according to the relevant educational program)

Necessary disciplines: " Programming. Part 1. Programming", "Data Structures and Algorithms".

3. Content of the academic discipline

Topic 1. HTML.
Topic 2. CSS.
Topic 3. LESS.
Topic 4. SASS.
Topic 5. SSSS.
Topic 6. Bootstrap.
Topic 7. JavaScript.
Topic 8. TypeScript.
Topic 9. jQuery.
Topic 10. React.
Topic 11. Virtual DOM.
Topic 12. Redux.
Topic 13. Angular.
Topic 14. Vue.

4. Educational materials and resources

Basic :

1. *Karaeva, N. V. WEB and mobile technologies in the economy [Electronic resource]: lecture notes / N. V. Karaeva; NTUU "KPI named after Igor Sikorsky". – Electronic text data (1 file: 2.02 MB). - Kyiv: NTUU "KPI named after Igor Sikorsky", 2017. - 158 p. – Title from the screen. https://ela.kpi.ua/handle/123456789/18947*
2. *User interface programming technologies (Front-end). Methodical instructions for performing laboratory work for bachelor's degree holders in the majors: 121 Software engineering, 123 Computer engineering, 126 Information systems and technologies of all forms of education / Compiled by: Aleschenko O.V. - K.: KPI named after Igor Sikorskyi, 2022. [Electronic resource]. Approved by the Department of Computing (Protocol No. 10 dated 05/25/2022). Agreed by the Methodical Commission of FIOT (protocol No. 10 dated 09.06.2022).*

https://comsys.kpi.ua/metodichni-vkazannya-po-disciplinam.

Additional :

3. *Marijn Haverbeke. Eloquent JavaScript, 3rd Edition: A Modern Introduction to Programming — No Starch Press, 2018. — 472 p. Also available online https://eloquentjavascript.net/*
4. *David Flanagan. JavaScript: The Definitive Guide: Master the World's Most-Used Programming Language. — 7th Edition — O'Reilly Media, 2020. — 706 p.*
5. *Boris Cherny. Programming TypeScript: Making Your JavaScript Applications Scale. — O'Reilly Media, 2019. — 324 p.*
6. *Aristeidis Bampakos, Pablo Deeleman. Learning Angular: A no-nonsense beginner's guide to building web applications with Angular 10 and TypeScript. — 3rd Edition — Packt Publishing, 2020. — 430 p.*
7. *Alex Banks, Eve Porcello. Learning React: Modern Patterns for Developing React Apps. — 2nd Edition O'Reilly Media, 2020. — 310 p.*
8. *Heitor Ramon Ribeiro. Vue.js 3 Cookbook: Discover actionable solutions for building modern web apps with the latest Vue features and TypeScript. — Packt Publishing, 2020. — 562 p.*

Educational content

5. Methods of mastering an educational discipline (educational component)

Full-time education

Names of sections and topics	*Number of hours*				
	In total	*including*			
		Lectures	*Practical (seminar)*	*Laboratory work*	*SRS*
Topic 1. HTML.	11	4	-	3	4
Topic 2. CSS.	11	4	-	3	4
Topic 3. SASS.	5	1	-	-	4
Topic 4. SSSS.	5	1	-	-	4
Topic 5. LESS.	5	1	-	-	4
Topic 6. Bootstrap.	5	1	-	-	4
Topic 7. JavaScript.	11	4	-	3	4
Topic 8. TypeScript.	5	1	-	-	4
Topic 9. jQuery.	5	1	-	-	4
Topic 10. React.	11	4	-	3	4
Topic 11. Virtual DOM.	6	2	-	-	4
Topic 12. Redux.	6	2	-	-	4
Topic 13. Angular.	14	6	-	3	5
Topic 14. Vue.	12	4	-	3	5
MCW	4	-	-	-	4
Test	4	-	-	-	4
Total in the semester:	120	36	-	18	66

Correspondence form teaching

Names of sections and topics	*Number of hours*				
	In total	*including*			
		Lecture s	*Practical (seminar)*	*Laboratory work*	*SRS*
Topic 1. HTML.	8	1	-	1	6
Topic 2. CSS.	8	1	-	1	6
Topic 3. SASS.	6.25	0,25	-	-	6
Topic 4. SSSS.	6.25	0,25	-	-	6
Topic 5. LESS.	6.25	0,25	-	-	6
Topic 6. Bootstrap.	6.25	0,25	-	-	6
Topic 7. JavaScript.	8	1	-	1	6
Topic 8. TypeScript.	6.25	0,25	-	-	6
Topic 9. jQuery.	7.25	0,25	-	-	7
Topic 10. React.	10	1	-	2	7
Topic 11. Virtual DOM.	7.25	0,25	-	-	7
Topic 12. Redux.	7.25	0,25	-	-	7
Topic 13. Angular.	10	1	-	2	7
Topic 14. Vue.	9	1	-	1	7
MCW	7	-	-	-	7
Test	7	-	-	-	7
Total in the semester:	120	8	-	8	104

Lecture classes

Full-time education

lecture no	*The title of the lecture topic, the list of main questions and tasks for the SRS*
1	***"HTML".*** *General characteristics of the HTML language. Basics of HTML syntax.* ***SRS:*** *Partially or completely complete the first laboratory work of this credit module using the HTML language.*
2	***"HTML. Continuation".*** *HTML5 elements and attributes. Global attributes. User attributes. HTML5 document structure and validation. Meta tags.* ***CCS:*** *Create an HTML5 document using (among other things) global attributes and meta tags. Validate the created document.*
3	***"CSS".*** *Cascading style sheets. Attaching styles to the page. Selectors. With SS in grace.* ***SRS:*** *Partially or completely complete the second laboratory work of this credit module using C SS technology.*
4	***"CSS. Continuation".*** *Current work on CSS specifications and how to get involved. Pseudo-classes and pseudo-elements. Modal window.* ***SRS:*** *Try to implement modality and use pseudo-classes and pseudo-elements in the second laboratory work of this credit module.*
5	***"SASS. CCCC».*** *A scripting metalanguage. Increasing the level of code abstraction and simplifying CSS files. SASS and SCSS syntax. Embedded rules. Admixtures (mixins). Cycles. Inheritance.* ***SRS:*** *Try to rework the second laboratory of this credit module using technology SASS or SSSS.*

6	***« LESS. Bootstrap ".*** *A dynamic language of styles. Nested metalanguage. Variables, nested rules, mixins, operators, and functions. Bootstrap CSS and HTML templates for typography, forms, buttons, navigation and other interface components.* ***SRS:*** *Try to rework the second laboratory of this credit module using technology LESS and Bootstrap.*
7	***"JavaScript".*** *Basics of Java Script. _ Implementation of the ECMAScript standard. Variables and constants. Data types. Operations. Error handling. Working with the browser and BOM. Working with the DOM.* ***SRS:*** *Partially or completely complete the third laboratory work of this credit module in the JavaScript programming language.*
8	***JavaScript. _ Continuation ".*** *JSON. Promise, async and await. Ajax. Fetch API. Functional programming. Object-oriented programming. Additional areas of application of JS.* ***CRC:*** *Try creating a JavaScript application using promise, async and await, Ajax and Fetch API.*
9	***"TypeScript. jQuery ".*** *Basics of TypeScript. Variables and constants. Data types. Functions. Function type and arrow functions. jQuery selectors and filters. Manipulating jQuery elements. Effects and animations in jQuery.* ***SRC:*** *Try to rework the third lab in this credit module using TypeScript and the jQuery library.*
10	***"React".*** *React basics. Rendering elements. Components. Props. Events State. Management of class components. Component life cycle.* ***CRC:*** *Partially or fully complete the fourth lab of this credit module using React.*
11	***"React. Continuation ".*** *Hooks Management of functional components. Routing.* ***SRS:*** *Try to rework the fourth lab in this credit module using hooks and functional components.*
12	***"Virtual DOM".*** *ReactDOM library. Harmonization. Shadow DOM. React Fiber.* ***SRS:*** *Try to understand the algorithm for matching the virtual and real DOM and understand how exactly the keys of the list elements are involved in this.*
13	***" Redux ".*** *Storage. actions Action creators. Reducer.* ***SRC:*** *Attempt to redo the fourth lab in this credit module using Redux.*
14	***"Angular".*** *Basics of Angular. Components. Modules. Component styles and templates. Data binding. Interaction between components.* ***SRS:*** *Partially or fully complete the fifth lab of this credit module using Angular.*
15	***"Angular. Continuation ".*** *Binding to child component events. Component life cycle. Interaction between modules. Directives. Services and dependency injection.* ***SRS:*** *Try to add to the implementation of the fifth laboratory work within the framework of this credit module the use of methods of the life cycle of the component, interaction between modules, directives, services and dependency injection.*
16	***"Angular. Continuation ".*** *Work with forms. HTTP and interaction with the server. Routing. Pipes.* ***SRS:*** *Try to add to the implementation of the fifth laboratory work within the framework of this credit module the use of work with forms, HTTP and interaction with the server, routing and p ipes.*
17	***" Vue ".*** *Vue Basics 3. The Vue object. Data binding. Processing user input. Events Two-way binding. Refs and control of HTML elements.* ***CSC:*** *Complete or partially complete the sixth lab of this credit module using Vue.*
18	***"Vue. Continuation ".*** *Vue Lifecycle. Conditional rendering and working with arrays. Work with forms. Components. Slots. Routing.*

	SRS: *Try to add robots with forms, conditional rendering, work with arrays, components, slots and routing to the implementation of the sixth laboratory work within the framework of this credit module.*

External form of education

lecture no	*The title of the lecture topic, the list of main questions and tasks for the SRS*
1	***"HTML".*** *General characteristics of the HTML language. Basics of HTML syntax. HTML5 elements and attributes. Global attributes. User attributes. HTML5 document structure and validation. Meta tags.* ***SRS:*** *Partially or completely complete the first laboratory work of this credit module using the HTML language. Create an HTML5 document using (among other things) global attributes and meta tags. Validate the created document.*
	"CSS". *Cascading style sheets. Attaching styles to the page. Selectors. With SS in mercy. Current work on CSS specifications and how to get involved. Pseudo-classes and pseudo-elements. Modal window.* ***SRS:*** *Partially or completely complete the second laboratory work of this credit module using C SS technology. Try to implement modality and use pseudo-classes and pseudo-elements in the second laboratory work of this credit module.*
2	***"SASS. CCCC».*** *A scripting metalanguage. Increasing the level of code abstraction and simplifying CSS files. SASS and SCSS syntax. Embedded rules. Admixtures (mixins). Cycles. Inheritance.* ***SRS:*** *Try to rework the second laboratory of this credit module using technology SASS or SSSS.*
	« LESS. Bootstrap ". *A dynamic language of styles. Nested metalanguage. Variables, nested rules, mixins, operators, and functions. Bootstrap CSS and HTML templates for typography, forms, buttons, navigation and other interface components.* ***SRS:*** *Try to rework the second laboratory of this credit module using technology LESS and Bootstrap.*
	"JavaScript". *Basics of Java Script. _ Implementation of the ECMAScript standard. Variables and constants. Data types. Operations. Error handling. Working with the browser and BOM. Working with the DOM. JSON. Promise, async and await. Ajax. Fetch API. Functional programming. Object-oriented programming. Additional areas of application of JS.* ***SRS:*** *Partially or fully complete the third laboratory work of this credit module in the JavaScript programming language. Try creating an application in the JavaScript programming language using promises, async and await, Ajax and Fetch API.*
3	***"TypeScript. jQuery ".*** *Basics of TypeScript. Variables and constants. Data types. Functions. Function type and arrow functions. jQuery selectors and filters. Manipulating jQuery elements. Effects and animations in jQuery.* ***SRC:*** *Try to rework the third lab in this credit module using TypeScript and the jQuery library.*
	"React". *React basics. Rendering elements. Components. Props. Events State. Management of class components. Component life cycle. Hooks Management of functional components. Routing.* ***CRC:*** *Partially or fully complete the fourth lab of this credit module using React. Try to redo the fourth laboratory work within the framework of this credit module using hooks and functional components.*
	"Virtual DOM". *ReactDOM library. Harmonization. Shadow DOM. React Fiber.*

	SRS: *Try to understand the algorithm for matching the virtual and real DOM and understand how exactly the keys of the list elements are involved in this.*
	" Redux ". *Storage. actions Action creators. Reducer.* ***SRC:*** *Attempt to redo the fourth lab in this credit module using Redux.*
4	***"Angular".*** *Basics of Angular. Components. Modules. Component styles and templates. Data binding. Interaction between components. Binding to child component events. Component life cycle. Interaction between modules. Directives. Services and dependency injection. Work with forms. HTTP and interaction with the server. Routing. Pipes.* ***SRS:*** *Partially or fully complete the fifth lab of this credit module using Angular. Try to add to the implementation of the fifth laboratory work within the framework of this credit module the use of component life cycle methods, interaction between modules, directives, services and dependency injection. Try to add to the implementation of the fifth laboratory work within the framework of this credit module the use of work with forms, HTTP and interaction with the server, routing and p ipes.*
	" Vue ". *Vue Basics 3. The Vue object. Data binding. Processing user input. Events Two-way binding. Refs and control of HTML elements. Vue Lifecycle. Conditional rendering and working with arrays. Work with forms. Components. Slots. Routing.* ***CSC:*** *Complete or partially complete the sixth lab of this credit module using Vue. Try to add working with forms, conditional rendering, working with arrays, components, slots and routing to the implementation of the sixth laboratory work within the framework of this credit module.*

Laboratory classes

Full-time education

No. z/p	*The name of the laboratory work*	*Number of aud. hours*
1	*HTML (Topic 1)*	*3*
2	*CSS (Topic 2)*	*3*
3	*JavaScript (Topic 7)*	*3*
4	*React (Topic 10)*	*3*
5	*Angular (Topic 13)*	*3*
6	*Vue (Topic 14)*	*3*
	Together:	*18*

External form of education

No. z/p	*The name of the laboratory work*	*Number of aud. hours*
1	*HTML (Topic 1)*	*1*
2	*CSS (Topic 2)*	*1*
3	*JavaScript (Topic 7)*	*1*
4	*React (Topic 10)*	*2*
5	*Angular (Topic 13)*	*2*
6	*Vue (Topic 14)*	*1*
	Together:	*8*

6. Independent work of student

Full-time education

No. z/p	*The name of the topic submitted for independent processing*	*Number of hours of SRS*
1	***"HTML".*** *Partially or completely complete the first laboratory work of this credit module using the HTML language.*	*2*
2	***"HTML. Continuation".*** *Create an HTML5 document using (among other things) global attributes and meta tags. Validate the created document.*	*2*
3	***"CSS".*** *Partially or completely perform the second laboratory work of this credit module using C SS technology.*	*2*
4	***"CSS. Continuation".*** *Try to implement modality and use pseudo-classes and pseudo-elements in the second laboratory work of this credit module.*	*2*
5	***"SASS. CCCC».*** *Try to rework the second laboratory of this credit module using technologies SASS or SSSS.*	*8*
6	***« LESS. Bootstrap ".*** *Try to rework the second laboratory of this credit module using technologies LESS and Bootstrap.*	*8*
7	***"JavaScript".*** *Partially or completely complete the third laboratory work of this credit module in the JavaScript programming language.*	*2*
8	***JavaScript. _ Continuation ".*** *Try creating an application in the JavaScript programming language using promises, async and await, Ajax and Fetch API.*	*2*
9	***"TypeScript. jQuery ".*** *Try to rework the third laboratory work within the framework of this credit module using the TypeScript language and the jQuery library.*	*8*
10	***"React".*** *Partially or fully complete the fourth lab of this credit module using React.*	*2*
11	***"React. Continuation ".*** *Try to redo the fourth laboratory work within the framework of this credit module using hooks and functional components.*	*2*
12	***"Virtual DOM".*** *Try to understand the algorithm for matching the virtual and real DOM and understand how exactly the keys of the list elements take part in this.*	*4*
13	***" Redux ".*** *Try to redo the fourth lab in this credit module using Redux.*	*4*
14	***"Angular".*** *Partially or fully complete the fifth lab of this credit module using Angular.*	*3*
15	***"Angular. Continuation ".*** *Try to add to the implementation of the fifth laboratory work within the framework of this credit module the use of component life cycle methods, interaction between modules, directives, services and dependency injection.*	*1*
16	***"Angular. Continuation ".*** *Try to add to the implementation of the fifth laboratory work within the framework of this credit module the use of work with forms, HTTP and interaction with the server, routing and p ipes.*	*1*
17	***" Vue ".*** *Complete part or all of the sixth lab of this credit module using Vue.*	*3*
18	***"Vue. Continuation ".*** *Try to add work with forms, conditional rendering, work with arrays, components, slots, and routing to the implementation of the sixth laboratory work within the framework of this credit module.*	*2*
19	***"Preparation for Modular control work (MCW)"***	*4*

20	***"Preparation for the test"***	*4*
	Together:	*66*

External form of education

No. z/p	*The name of the topic submitted for independent processing*	*Number of hours of SRS*
1	***"HTML".*** *General characteristics of the HTML language. Basics of HTML syntax. HTML5 elements and attributes. Global attributes. User attributes. HTML5 document structure and validation. Meta tags.* ***SRS:*** *Partially or completely complete the first laboratory work of this credit module using the HTML language. Create an HTML5 document using (among other things) global attributes and meta tags. Validate the created document.*	*6*
2	***"CSS".*** *Cascading style sheets. Attaching styles to the page. Selectors. With SS in mercy. Current work on CSS specifications and how to get involved. Pseudo-classes and pseudo-elements. Modal window.* ***SRS:*** *Partially or completely complete the second laboratory work of this credit module using C SS technology. Try to implement modality and use pseudo-classes and pseudo-elements in the second laboratory work of this credit module.*	*6*
3	***"SASS. CCCC».*** *A scripting metalanguage. Increasing the level of code abstraction and simplifying CSS files. SASS and SCSS syntax. Embedded rules. Admixtures (mixins). Cycles. Inheritance.* ***SRS:*** *Try to rework the second laboratory of this credit module using technology SASS or SSSS.*	*12*
4	***« LESS. Bootstrap ".*** *A dynamic language of styles. Nested metalanguage. Variables, nested rules, mixins, operators, and functions. Bootstrap CSS and HTML templates for typography, forms, buttons, navigation and other interface components.* ***SRS:*** *Try to rework the second laboratory of this credit module using technology LESS and Bootstrap.*	*12*
5	***"JavaScript".*** *Basics of Java Script. _ Implementation of the ECMAScript standard. Variables and constants. Data types. Operations. Error handling. Working with the browser and BOM. Work with DOM. JSON. Promise, async and await. Ajax. Fetch API. Functional programming. Object-oriented programming. Additional areas of application of JS.* ***SRS:*** *Partially or fully complete the third laboratory work of this credit module in the JavaScript programming language. Try creating an application in the JavaScript programming language using promises, async and await, Ajax and Fetch API.*	*6*
6	***"TypeScript. jQuery ".*** *Basics of TypeScript. Variables and constants. Data types. Functions. Function type and arrow functions. jQuery selectors and filters. Manipulating jQuery elements. Effects and animations in jQuery.* ***SRC:*** *Try to rework the third lab in this credit module using TypeScript and the jQuery library.*	*13*
7	***"React".*** *React basics. Rendering elements. Components. Props. Events State. Management of class components. Component life cycle. Hooks Management of functional components. Routing.*	*7*

	CRC: *Partially or fully complete the fourth lab of this credit module using React. Try to redo the fourth laboratory work within the framework of this credit module using hooks and functional components.*	
8	***"Virtual DOM".*** *ReactDOM library. Harmonization. Shadow DOM. React Fiber.* ***SRS:*** *Try to understand the algorithm for matching the virtual and real DOM and understand how exactly the keys of the list elements are involved in this.*	*7*
9	***" Redux ".*** *Storage. actions Action creators. Reducer.* ***SRC:*** *Attempt to redo the fourth lab in this credit module using Redux.*	*7*
10	***"Angular".*** *Basics of Angular. Components. Modules. Component styles and templates. Data binding. Interaction between components. Binding to child component events. Component life cycle. Interaction between modules. Directives. Services and dependency injection. Work with forms. HTTP and interaction with the server. Routing. Pipes.* ***SRS:*** *Partially or fully complete the fifth lab of this credit module using Angular. Try to add to the implementation of the fifth laboratory work within the framework of this credit module the use of component life cycle methods, interaction between modules, directives, services and dependency injection. Try to add to the implementation of the fifth laboratory work within the framework of this credit module the use of work with forms, HTTP and interaction with the server, routing and pipes.*	*7*
11	***" Vue ".*** *Vue Basics 3. The Vue object. Data binding. Processing user input. Events Two-way binding. Refs and control of HTML elements. Vue Lifecycle. Conditional rendering and working with arrays. Work with forms. Components. Slots Routing.* ***CSC:*** *Complete or partially complete the sixth lab of this credit module using Vue. Try to add work with forms, conditional rendering, work with arrays, components, slots, and routing to the implementation of the sixth laboratory work within the framework of this credit module.*	*7*
12	***"Preparation for Modular control work (MCW)"***	*7*
13	***"Preparation for the test"***	*7*
	Together:	*104*

Policy and control

7. Policy of academic discipline (educational component)

- rules for the protection of laboratory work:
  - the student provides the program code for laboratory work,
  - the student defends the provided software code through an interview;
- deadline and rescheduling policy:
  - the student has the right to submit and resubmit laboratory work up to and including the day of assessment,
  - there is no limit on the number of folds;
- academic integrity policy:
  - the student has the right to protect the program code that was written not only by him, but in this case the student must understand "what" and "how" this code performs, be able to use it as a basis for making modifications that may be suggested by the teacher.

8. Types of control and rating system for evaluating learning outcomes (RSO)

Types of control from the educational discipline "Technologies of programming user interfaces (Front-end)" include:

Laboratory works:

Independent performance of six laboratory works is planned. The topics of laboratory works are coordinated in time and content with the topics of lectures. Carrying out laboratory work in full allows you to acquire practical skills in programming user interfaces.

Current control:

It is planned to carry out control work (MKR)

Semester control:

Assessment is conducted in the form of an interview with the student to objectively determine the level of knowledge, skills and practical skills acquired during the semester

The student's semester rating consists of the points he receives for the types of work in accordance with Table 1.

Table 1

Assessment of individual types of student academic work (in points)

Type of educational work		***Total by type of work***
Performance and protection of laboratory work No. 1		*15*
Performance and protection of laboratory work No. 2		*15*
Performance and protection of laboratory work No. 3		*15*
Performance and protection of laboratory work No. 4		*15*
Performance and protection of laboratory work No. 5		*15*
Performance and protection of laboratory work No. 6		*15*
MCW	***Rk***	*10*
	Rp	***100***
	Credit (R3) as desired	***20***

The individual current rating of the student (***Rp*** *) consists of the points he receives for performing laboratory work (* ***Rl*** *) and MKR* ***(*** ***Rk*** *). During the semester, students perform 6 laboratory works. The maximum number of points for each laboratory work is 15. Points are awarded for:*

- *theoretical component – 7 points,*
- *practical component - 8 points.*

The maximum possible score for laboratory work is 15 points. The maximum number of points for all laboratory works is 6 x 15 = 90 points.

Calculation of the scale size (R) of the rating.

The sum of the weighted points of control measures during the semester is:

Rp = Rl + Rk, *where Rp is the student's semester rating (MCW, laboratory work).*

A necessary condition for a student's admission to credit is his individual semester rating (Rp), not less than 40 points, and the absence of arrears from laboratory work and MKR. If the mentioned requirements are not met, the student will not be admitted to the credit.

Table of correspondence of rating points to grades on the university scale:

Scores	***Rating***
100-95	*Perfectly*
94-85	*Very good*
84-75	*Fine*
74-65	*Satisfactorily*
64-60	*Enough*
Less than 60	*Unsatisfactorily*
Admission conditions not met	*Not allowed*

9. Additional information on the discipline (educational component)

As part of the study of the discipline "Technology of programming user interfaces (Front-end)", crediting of points obtained as a result of distance courses is allowed, provided that the program of this course is agreed with the teacher.

Working program of the academic discipline (syllabus):
Designed *by Oleksii Aleshchenko, a senior lecturer at the Department of Computer Engineering*
Adopted *by the Department of Computer Engineering (Protocol No. 10 dated 25 May 2022)*
Approved *by the Methodical Commission of the faculty (protocol No. 10 dated 09 June 2022)*