National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"

Department of Computer Engineering

Server software development technologies (backend)

The program of the academic discipline (Syllabus)

1. Details of the academic discipline	
Cycle of Higher Education	***First cycle of higher education (Bachelor's degree)***
Field of Study	*12 Information technologies*
Speciality	*121 Software Engineering* *123 Computer engineering*
Education Program	*Computer Systems Software Engineering* *Computer Systems and Networks*
Type of Course	*Selective*
Mode of Studies	*Full-time education*
Year of studies, semester	*3 year (1 semester)*
ECTS workload	*4 credits (120 hours)* *Full-time education: Lectures 18 (36), Laboratory 9 (18), Independent work (66)* *Extramural studies: Lectures 18 (8), Laboratory 9 (8), Independent work (104)*
Testing and assessment	*Semester credit*
Course Schedule	*According to the schedule for the autumn semester of the current academic year at http://rozklad.kpi.ua*
Language of Instruction	*Ukrainian*
Course Instructors	Lecturer: Volodymyr Valko, *valko.volodya@gmail.com* Laboratory: Volodymyr Valko, *valko.volodya@gmail.com*
Access to the course	Google Workspace for Education code: https://classroom.google.com/c/NDg5Mzc5ODM4MTc1?cjc=7cmg2mo

2. Program of academic discipline

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

The goal of the discipline "Server software development technologies (backend)" is to study the theoretical foundations, principles of building server software and acquire practical skills in building such projects. The subject of study of the credit module "Technology of server software development (backend)" is the architecture and functionality of modern server software; interaction with databases on the server; features of modern server software programming and their support.

The discipline enhances the following general and professional competencies:

- ЗК2 - ability to learn and acquire modern knowledge;
- ФК1 - ability to apply the legislative and regulatory framework, as well as national and international requirements, practices and standards in order to carry out professional activities in the field of computer engineering.
- ФК2 - ability to use modern methods and programming languages for development of algorithms and software.
- ФК5 - ability to use the tools and systems of design automation to development of components of computer systems and networks, Internet applications, cyber-physical systems, etc.
- ФК9 - ability to systematically support, use, adapt and operate existing information technologies and systems.

- ФК13 - ability to solve problems in the field of computer and information technologies, determine the limitations of these technologies.
- ФК14 - ability to design systems and their components taking into account all their aspects. life cycle and task, including creation, configuration, operation, maintenance and disposal.

In accordance with the above, strengthened general and professional competencies will provide the following learning outcomes:

- knowledge of fundamental concepts, paradigms and basic principles of server software operation;
- the ability to use theoretical, logical and architectural foundations for the development of server lost support and the ability to apply them when solving professional tasks;
- ability to develop server software for typical areas of use, including, using versioning systems and modern frameworks;
- ability to develop and use drivers of architecture-dependent elements based on knowledge of general principles of organization and functioning of embedded systems;
- the ability to participate in team work on the design of server software;
- the ability to form and provide requirements for the reliability of server software in accordance with customer requirements, specifications and standards.

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

When studying the discipline "Server software development technologies (backend)", it is advisable to use the knowledge obtained during the study of previous disciplines: "Programming", "Object-oriented programming", "Algorithms and calculation methods", "System programming", "Structures data and algorithms", "Discrete mathematics", "Software engineering".

The discipline is basic for courses from the F-catalogue of optional disciplines included in the certificate program: "Server software development technologies (backend)" (selective), "Technologies of designing intelligent systems" (selective), "Technologies of testing (QA) of embedded systems" (selective), "Infrastructure IT project management" (selective), and is also useful for studying the regulatory disciplines "Organization of databases", "System software. Course work", "System programming".

3. Structure of the credit module

Introduction. The role of version control systems in software development
Section 1. The C Preprocessor
- Topic 1.1. VCS
- Topic 1.2. How git works
- Topic 1.3. Basic operations in git. Advanced use of git

Section 2. Python as a server software programming language. Theory
- Topic 2.1. Python Interpreter
- Topic 2.2. Representation of objects in memory
- Topic 2.3. Types in Python
- Topic 2.4. OOP in Python
- Topic 2.5. Polymorphism
- Topic 2.6. Incapsulation
- Topic 2.7. Inheritance
- Topic 2.8. Magic methods in Python

Section 3. Python as a server software programming language. Practice
- Topic 3.1. Basic operations
- Topic 3.2. Peculiarities of functions in python
- Topic 3.3. Features of OOP in python in practice
- Topic 3.4. Implementation of decorators

Section 4. Frameworks, APIs

Topic 4.1. Framework definition
Topic 4.2. Work with JSON
Topic 4.3. REST API definition
Topic 4.4. Basic principles of working with Docker
Topic 4.5. Basic principles of working with docker-compose
Topic 4.6. Library definition
Topic 4.7. Practice with Docker

Section 5. Databases

Topic 5.1. Database definition
Topic 5.2. Main types of databases and areas of use
Topic 5.3. Work with PostgreSQL
Topic 5.4. Features of transactions in PostgreSQL
Topic 5.5. Blocking in PostgreSQL
Topic 5.6. Features of join in PostgreSQL
Topic 5.7. Indices in PostgreSQL
Topic 5.8. Replication in PostgreSQL

Section 6. Authentication and authorization

Topic 6.1. Definition of authentication
Topic 6.2. Definition of authorization
Topic 6.3. Multi-factor authorization
Topic 6.4. Roles

Section 7. Authentication standards and frameworks

Topic 7.1. JWT definition
Topic 7.2. Structure of JWT
Topic 7.3. OAuth2.

Section 8. Server software security

Topic 8.1. The concept of software vulnerability
Topic 8.2. Main types of vulnerabilities
Topic 8.3. Detailed review of the most common vulnerabilities - CSRF, XSS, SQLi

4. Educational resources and materials

Basic literature

1. Fundamentals of programming. Python. Part 1 [Electronic resource]: textbook for students. specialty 122 "Computer science", specialization "Information technologies in biology and medicine" / A. V. Yakovenko; KPI named after Igor Sikorsky. – Electronic text data (1 file: 1.59 MB). – Kyiv: KPI named after Igor Sikorskyi, 2018. – 195 p.
2. Fundamentals of programming in the algorithmic language Python [Electronic resource]: teaching. manual for students of the educational program "Computer systems and networks" specialty 123 "Computer engineering" / M.A. Novotarskyi - Electronic text data (1 file: 18 MB). – Kyiv: KPI named after Igor Sikorskyi, 2022. – 701 p.
3. Databases and management tools. Practicum [Electronic resource]: education. manual for students specialty 123 - Computer engineering. / V.I. Pavlovsky, A.V. Petrashenko, D.V. Victory; KPI named after Igor Sikorsky. – Electronic text data (1 file: 7.7 MB). – Kyiv: KPI named after Igor Sikorskyi, 2021. – 112 p.

Additional literature

4. L.Ramalho (2015) Fluent Python: Clear, Concise, and Effective Programming 1st edition, O`Reilly Media.
5. David Beazley, Brian K. Jones (2013) Python Cookbook, 3rd edition, O`Reilly Media.
6. Zed Shaw (2013) Learn Python the Hard Way, 3rd ed. Addison-Wesley Professional

Information resources

1. Online Course on “Server software development technologies (backend)” via platform "Sikorsky" using Google Workspace for Education:

https://classroom.google.com/c/NDg5Mzc5ODM4MTc1?cjc=7cmg2mo

2. Recordings of lectures – YouTube. – https://www.youtube.com/playlist?list=PLDBW1r7eLmO96r_fzoF3QYI0fRGoDtBpw

5. Laboratory work

The purpose of the laboratory work is to acquire skills and practical application of the principles of designing and developing server software. Software development tools (vim, Visual Studio Code, PyCharm, Docker, docker-compose, git) are used to perform laboratory work.

Topics of laboratory works:

Laboratory work 1. Working with git.
Laboratory work 2. Setting up the environment, Development of the basic REST API
Laboratory work 3. Validation, error handling, work with ORM.
Laboratory work 4. Implementation of authentication, presentation of the project.

6. Independent work of the student

Types of independent work (66 hours):

- preparation for classroom classes (0,5 hours x 18 lectures = 9 hours);
- performance of an individual task for laboratory work, solving problems, drawing up a protocol, placing the results on GitHub (2 hours x 8 laboratory works = 16 hours);
- execution of module's tests (4 hours x 2 Module's Tests = 8 hours);
- preparation for express tests (4 hours);
- working out topics for independent work, downloading and setting up software for laboratory work (29 hours).

6.1. Topics for self-study (Full-time education)

Section 2. Python as a server software programming language. Theory
Topic 2.3. Types in Python
Topic 2.5. Polymorphism
Topic 2.6. Incapsulation
Section 4. Frameworks, APIs
Topic 4.2. Work with JSON
Topic 4.4. Basic principles of working with Docker
Topic 4.5. Basic principles of working with docker-compose
Section 5. Databases
Topic 5.1. Database definition
Topic 5.2. Main types of databases and areas of use
Topic 5.8. Replication in PostgreSQL

7. The method of teaching the discipline in the extramural studies

The content of lectures and self-study

Section 2. Python as a server software programming language. Theory
Topic 2.1. Python Interpreter
Topic 2.2. Representation of objects in memory
Topic 2.3. Types in Python
Topic 2.4. OOP in Python

Topic 2.5. Polymorphism
Topic 2.6. Incapsulation
Topic 2.7. Inheritance
Topic 2.8. Magic methods in Python

Topics for self-study:

Topic 1.3. Basic operations in git. Advanced use of git

Lection 2. Databases

Topic 5.1. Database definition
Topic 5.2. Main types of databases and areas of use
Topic 5.3. Work with PostgreSQL
Topic 5.4. Features of transactions in PostgreSQL
Topic 5.5. Blocking in PostgreSQL
Topic 5.6. Features of join in PostgreSQL

Topics for self-study:

Topic 5.7. Indices in PostgreSQL
Topic 5.8. Replication in PostgreSQL

Lection 3. Authentication and authorization

Topic 6.1. Definition of authentication
Topic 6.2. Definition of authorization
Topic 6.3. Multi-factor authorization
Topic 6.4. Roles

Lection 4. Authentication standards and frameworks

Topic 7.1. JWT definition
Topic 7.2. Structure of JWT
Topic 7.3. OAuth2.

Topics for self-study:

Topic 8.2. Main types of vulnerabilities
Topic 8.3. Detailed review of the most common vulnerabilities - CSRF, XSS, SQLi

Subjects of laboratory works for self-fulfillment

Laboratory work 1. Working with git.
Laboratory work 2. Setting up the environment, Development of the basic REST API
Laboratory work 3. Validation, error handling, work with ORM.
Laboratory work 4. Implementation of authentication, presentation of the project.

Subjects of laboratory work for classroom work

Classroom classes 1:

Laboratory work 1. Creating a test project, working with basic operations in git — commit, push, merge. Creating tags, getting to know GitHub actions

Classroom classes 2:

Defense of laboratory works

Classroom classes 3:

Defense of laboratory works

Classroom classes 4:

Defense of laboratory work 4 and presentation of the entire project implemented in the laboratory

course

Types of independent work for correspondence education (104 hours):

- preparation for classroom classes (1,5 hours x 4 lectures = 6 hours);
- preparation for express tests (4 hours);
- self-fulfillment of laboratory works 1-5 (3,5 hours x 5 laboratory works = 17,5 hours);
- preparation for laboratory works (6 – 8), drawing up a protocol, placing the results on GitHub (1,5 hours x 3 laboratory works = 4,5 hours);
- execution of module's tests (4 hours x 2 Module's Tests = 8 hours);
- preparation for the semester credit (4 hours);
- working out topics for independent work, downloading and setting up software for laboratory work (60 hours).

3. Policy and Assessment

8. Course policy

The grade that a student can receive for each laboratory work and for each module's test is given in table 1 of semester work evaluations, chapter 8 of the syllabus.

Completion of all laboratory works is mandatory for admission to the semester credit. The condition of admission to the semester credit is the enrollment of all laboratory works and a starting rating of at least 30 points.

Deadlines are set for laboratory work. Performance of laboratory work outside the set time is accompanied by penalty points, which are deducted from the grade for the protocol.

Penalty points and hard deadlines are not introduced during martial law.

Module's Test is performed independently according to an individual task, Module's Test is not accepted beyond the set deadline. Module's Test are not rewritten in case of a negative grade, a negative grade for the Module's Test (less than 9 points (<60%)) is equal to 0 points, in this case the Module's Test is not counted.

Individual lecture topics are accompanied by short express tests (for 20 minutes), which include the material of the studied topic and questions that are asked for independent study. The points obtained for the test are included in the semester rating. Current tests are not retaken.

Types of independent work for Thus, the minimum grade that a student can receive to qualify for a course = 60 points, the maximum is 100 points for the completion of all types of tasks during the semester.

Students who have fulfilled all admission requirements (completed all laboratory works) and have a rating of less than 60 points, as well as applicants who wish to improve their rating, have the opportunity to take the semester test at the last class on the schedule.

In the case of performance of the semester test, the rating is defined as the sum of points for semester test and points for individual semester tasks.

The individual work of the student is related to the performance of laboratory work. The maximum number of points for individual work per semester is 60 points. The maximum mark for the test is 40 points. In this way, the student has the opportunity to increase his rating by writing a semester test and adding additional points to the number of points received for individual work during the semester.

After completion of the semester test, if the grade for the semester test is higher than the rating, the student receives a grade based on the results of the semester test. If the grade for the semester test is lower than the rating, the student's previous rating (with the exception of points for the individual task) is canceled and he receives a grade based on the results of the semester test. This option forms a responsible attitude of the student towards making a decision on the completion of the semester test, forces him to critically assess the level of his training and carefully prepare for the credit.

9. Monitoring and grading policy

Distribution of study time by types of classes and tasks in the discipline according to the working study plan. The credit module is allocated 120 hours and 4 credits.

The student's semester rating from the credit module is calculated based on a 100-point scale. The rating consists of points that the student receives for performing 8 laboratory works R_{LW}, two Module's Tests R_{MT} and two express tests R_{ET}.

The maximum number of points for all laboratory works is 60 points, i.e. $R_{LW} = 60$.

The criteria for evaluating laboratory works are as follows:

- timeliness of drawing up the protocol for the laboratory work, completeness of the theoretical or practical task in the protocol, its timely upload: 0 - 2 points;
- correct functioning of the developed models on software or hardware, demonstration of own repository on GitHub with materials of performed laboratory work: 0 - 4 points;
- survey on the topic of laboratory work for crediting the practical part of the work, defense of the results obtained in the work, answers to additional theoretical questions of the teacher: 0 - 2 points.

The maximum number of points per Module's Test $R_{MT} = 2 \times 15 = 30$ points.

The mark for Module's Test decreases by:

- incorrect design of work on the remote repository, errors in the build files;
- lack of comments in the program code and design of algorithms;
- absence of comments and explanations during calculations.

The maximum number of points for express tests $R_{ET} = 2 \times 5 = 10$ points, the tests are conducted in the form of automated testing.

Table 1. Detailing of points for current works for the semester

Task name	Form of control	Number of points	Admission to automatic credit	Total points
Laboratory work 1	Task completed	10	5	15
	Answers on questions	3		
	Protocol	2		
Laboratory work 2	Task completed	10	5	15
	Answers on questions	3		
	Protocol	2		
Laboratory work 3	Task completed	10	5	15
	Answers on questions	3		
	Protocol	2		
Laboratory work 4	Task completed	10	5	15
	Answers on questions	3		
	Protocol	2		
Individual work	-	-	-	60
Express tests	2 x 5	10	5	10
Module's Test	Module's Test 1	15	9	15
	Module's Test 2	15	9	15
Total points		100	60	100

The maximum number of points for credit work is equal to $R_C = 40$ points.

The credit card contains 4 tasks (one theoretical and three practical) on the subject of lectures and laboratory works performed during the semester. Each question is evaluated from 0 to 10 points.

Evaluation criteria for each question at four levels:

- correct and meaningful answer: 9 – 10 points;
- correct answer, incomplete explanations: 6 – 8 points;
- the answer contains errors: 3 – 5 points;

- no answer or the answer is incorrect: 0 points.

The grade for the semester test is calculated according to table 2. An unsatisfactory grade does not give the right to credit additional points. Additional points to the semester grade are also not credited if the grade for semester test is lower than the student's current semester grade (according to Table 2). The semester grade can be increased to the minimum number of points of the semester grade (see Table 3), which corresponds to the grade received for the semester test (see Table 2).

Table 2. Determining the grade for credit work

Score	*Grade*
40-38	Excellent
37-34	Very good
33-30	Good
29-26	Satisfactory
25-24	Sufficient
below 24	Not Graded

Calendar certification of students (for 8 and 14 weeks of semesters) in the discipline is carried out according to the value of the student's current rating at the time of certification. If the value of this rating is at least 50% of the maximum possible at the time of certification, the student is considered certified. Otherwise, "Not Graded" is displayed in the certification information.

A necessary condition for a student's admission to credit is the completion and defense of all laboratory work with a total of at least 30 points.

The number of points a student receives per semester is determined by the formula

$$RC = R_{LW} + R_{MT} + R_{ET}$$

The maximum number of points per semester does not exceed RC = 100.

Taking into account the received sum of points, the final grade is determined by table 3.

If a student writes a semester test, the number of points the student receives per semester is determined by the formula

$$RC = R_{LW} + R_{C}$$

The maximum number of points per semester does not exceed RC = 100.

Taking into account the received sum of points, the final grade is determined by table 3.

Table 3. Determination of the semester grade

Score	*Grade*
100-95	Excellent
94-85	Very good
84-75	Good
74-65	Satisfactory
64-60	Sufficient
below 60	Fail
Course requirements are not met	Not Graded

The program of the academic discipline (syllabus):

Made by assistant of the department of CE, Volodymyr Valko.

Approved by the Department of Computer Engineering (protocol No. 10 dated 05.25.2022).

Agreed by the methodical commission of FICT (protocol No. 10 dated 06.09.2022).