НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено:

Завідувач кафедри

_____ Олександр Коваль

«___»_____2020 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Геометричне моделювання в інформаційних системах»

спеціальності 122 «Комп'ютерні науки та інформаційні технології»

на тему: «Автоматизована система контролю працездатності ресурсів Приймальної комісії КПІ ім. Ігоря Сікорського »

Виконала:
студентка IV курсу, групи ТР-61
Горбатенко Ксенія Вікторівна _____

Керівник:
доцент,к.т.н
Залевська Ольга Валеріївна _____

Рецензент:
доцент,к.т.н
Олексій Дмитрович Фіногенов _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2020 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший рівень

Напрям підготовки 122 Комп’ютерні науки та інформаційні технології

Спеціалізація Геометричне моделювання в інформаційних системах

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр Коваль
(підпис)
”___”_____2020р.

ЗАВДАННЯ
на дипломну роботу студенту

_____Горбатенко Ксенії Вікторівні _____
(прізвище, ім’я, по батькові)

1.Тема роботи: Автоматизована система контролю працездатності ресурсів Приймальної комісії КПІ ім. Ігоря Сікорського
керівник роботи Залевська Ольга Валеріївна доцент,к.т.н
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від ”25” травня 2020р. № 1268-с

2. Строк подання студентом роботи _“16” червня 2020р._____

3. Вихідні дані до роботи : мова програмування Python , інтегроване середовище PyCharm.

4.Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):1)проаналізувати специфіку роботи Приймальної комісії КПІ ім. Ігоря Сікорського 2) проаналізувати існуючі програмні засоби, їх переваги та недоліки, засоби реалізації програмної системи 3)розробити програмний додаток для вирішення поставленої задачі та продемонструвати його роботу

5. Перелік ілюстративного матеріалу:
1) Актуальність; 2) Мета;3) Основні задачі;4) Структура приймальної комісії КПІ ім. Ігоря Сікорського;5) Ресурси Приймальної комісії;6) Топологія локальної мережі приймальної комісії КПІ ім. Ігоря Сікорського;7) Обгрунтування вибору мови програмування;8) Логічна структура ПД UML-діаграма;9) Програма–агент;
10) Діаграма прецедентів;11) Приклади виконання програми;12) Висновки
6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання ”___”_____201___ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Затвердження теми роботи	20.04.20	
2.	Вивчення та аналіз задачі	7.05.20 - 14.05.20	
3.	Розробка архітектури та загальної структури системи	15.05.20 - 17.05.20	
4.	Розробка структур окремих підсистем	18.05.20 - 19.05.20	
5.	Програмна реалізація системи	20.05.20 - 24.05.20	
6.	Оформлення пояснювальної записки	25.05.20 - 11.06.20	
7.	Захист програмного продукту	12.06.20	
8.	Передзахист	12.06.20	
9.	Захист	16.06.20	

Студентка _____ _____Горбатенко К.В._____
(підпис) (прізвище та ініціали,)

Керівник роботи _____ _____Залевська О.В._____
(підпис) (прізвище та ініціали,

АНОТАЦІЯ

Метою дипломної роботи є розробка програмного продукту, який дозволить систематизувати, організувати та контролювати інформаційний потік Приймальної комісії КПІ ім. Ігоря Сікорського та надасть можливість прогнозування календарного плану виходу на работу співробітників приймальної та відбіркових комісій.

Користувачами системи є секретаріат Приймальної комісії закладу вищої освіти та відбіркові комісії факультетів та інститутів

Вхідною інформацією є внутрішня структура Приймальної комісії та статистичні дані попередніх років про подані заяви абітурієнтів.

Вихідною інформацією є рекомендації для Приймальної комісії щодо технічного та кадрового забезпечення ПК протягом вступної кампанії

Пояснювальна заиска містить аркушів, 33 ілюстрації, 5 додатків та 14 використаних джерел.

ABSTRACT

The purpose of the thesis is to develop a software product that will systematize, organize and control the information of the Admissions Committee KPI. Igor Sikorsky and will provide an opportunity to predict the calendar plan for the employment of selection and selection commissions.

The users of the system are the secretariat of the Admissions Committee of the higher education institution and the selection commissions of the faculties and institutes.

The input information is the internal structure of the Admissions Committee and statistics from previous years on the applications submitted entrants.

The initial information is the recommendations for the Admissions Committee on technical and personnel support of the PC during the introductory campaign

The explanatory note contains sheets, 33 illustrations, 5 appendices and 14 sources used.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ЄДЕБО – Єдиної державної електронної бази з питань освіти;

ПЗ – програмне забезпечення;

ПК – персональний комп'ютер;

ВК – відбіркова комісі

ВСТУП

Запорукою успішного проведення вступної кампанії до закладів вищої освіти є злагоджена, швидка та вчасна обробка інформації Приймальною комісією закладу.

В КПІ ім. Ігоря Сікорського до складу Приймальної комісії входять відбіркові комісії факультетів та інститутів , очолювані деканами та директорами. Також до складу Приймальної комісії входить секретаріат Приймальної комісії, який узгоджує, направляє та перевіряє своєчасну роботу відбіркових комісій, структурних підрозділів.

Аналіз роботи Приймальної комісії за 2017 - 2019 років виявив низку чинників, що впливають на результат проведення вступної кампанії. Несвоєчасне виконання своїх обов'язків одного зі структурних підрозділів призводило до затримки роботи всієї комісії. Оскільки, до складу відбіркових комісій факультетів/інститутів входило близько тисячі співробітників, то втрати часу були значними. Таким чином, говорити про автоматизацію роботи секретаріата для забезпечення вдалої вступної кампанії в очному режимі немає змоги, в зв'язку з цим, виникає питання про розробку програмного забезпечення, що дало б змогу контролювати та систематизувати ресурси приймальної комісії.

Наразі, таке програмне забезпечення відсутнє.

Для розробки ПЗ необхідно вирішити наступні задачі:

1). Провести аналіз інформаційного потоку Приймальної комісії протягом останніх років.

2). Проаналізувати комп'ютерне забезпечення відбіркових комісій факультетів

3). В залежності від функціональних можливостей комп'ютерного устаткування, кількості поданих заяв абітурієнтів розробити план виходу на работу технічних співробітників відбіркових комісій факультетів та кількості комп'ютерів, необхідних для своєчасної обробки інформації

1.ПОСТАНОВКА ЗАДАЧІ

Проаналізувати специфіку роботи Приймальної комісії КПІ ім. Ігоря Сікорського, з’ясувати класифікацію ресурсів підрозділу, розглянути існуючі програмні додатки для контролю їх працездатності а також вказати їх переваги та недоліки, розглянути засоби реалізації програмної системи контролю працездатності ресурсів Приймальної комісії КПІ ім. Ігоря Сікорського проаналізувати їх переваги та недоліки, описати та розробити основні функції програмної реалізації.

На вході в систему має бути файл, де зберігається інформація про факультети та дати подачі заявок абітурієнтами в ЄДЕБО,

На виході:

1)Статистика по всіх комп'ютерах за дату;

2)Статистика по всіх комп'ютерах одного факультету за дату;

3)Статистика по одному комп'ютеру за всі дати;

4)Робочі ресурси по всіх факультетах за дату;

5)Робочі ресурси по одному факультету за дату;

6)Наявність посилань на сайті.

Потенційними користувачами є співробітники приймальної комісії НТУУ «КПІ».

2. СПЕЦИФІКА РОБОТИ ПРИЙМАЛЬНОЇ КОМІСІЇ КПІ ім. Ігоря Сікорського

Підсумки результатів проведення вступних компанії КПІ ім. Ігоря Сікорського попередніх років свідчать, що якість роботи Приймальної комісії значною мірою забезпечується сумлінним виконанням своїх функціональних обов'язків відбірковими комісіями факультетів університету, які несуть важливу функцію прийому абітурієнтів.

Аналіз результатів роботи відбіркових комісій КПІ ім. Ігоря Сікорського показав деякі проблеми, які вносять суттєвий вплив на якість роботи.

Одним з важливих чинників є той факт, що щороку склади ВК частково або цілком змінюються (значною мірою секретаріату ВК), важливою є якісна підготовка співробітників до ефективного виконання функціональних обов'язків, які покладені на членів відбіркових комісій.

Положення про Приймальну комісію КПІ ім. Ігоря Сікорського [1] регламентує створення підрозділів, одними з яких є відбіркові комісії, для виконання повною мірою функцій та завдань приймальної комісії

Відбіркову комісію очолює голова відбіркової комісії. Також призначається заступник голови відбіркової комісії . Організаційно-технічне навантаження несе секретаріат відбіркової комісії: відповідальний секретар ВК та його заступники, а також члени відбіркової комісії. Це оператори та технічні секретарі. Їх призначають з числа викладачів або навчально-допоміжного персоналу факультетів, кафедр університету.

Обов'язками співробітників відбіркової комісії серед інших є: прийом документів, оформлення особових справ вступників [2].

Слід зазначити, що робота ПК може бути максимально ефективною в разі оптимального поєднання використання людських ресурсів - попередньої підготовки,

чіткого знання власних обов'язків з технічним потенціалом - можливостями комп'ютерів та програмного забезпечення. Отже, важливим є контроль ефективності поєднання людських ресурсів з можливостями технічного забезпечення.

Під час вступної кампанії вступники подають заяви через систему електронного вступу, що фіксуються в ЄДЕБО та обробляються відбірковими комісіями.

2.1. «Ресурси» приймальної та відбіркової комісії

Задля забезпечення відлагодженої безперебійної роботи приймальної комісії «КПІ ім.Ігоря Сікорського» застосовують допоміжні засоби (ресурси). Для реалізації контролю працездатності ресурсів, їх необхідно розбити по категоріях і дати точне визначення кожній. Схематичне зображення представлено на рисунку 1.

Рисунок 2.1 – Ресурси Приймальної комісії

2.1.1. Людські ресурси

Обробку заяв абітурієнтів забезпечують співробітники відбіркової комісії – оператори, вони виконують перевірку на відповідність даних поданих вступником умовам прийому та за потребою змінюють статус заяви. В даному випадку «людські ресурси» визначають кількість спеціалістів, яка необхідна для обробки заяв, що подані до підрозділу; кількість задіяних робочих місць, обладнаних справною технікою.

2.1.2. Ресурси ПК

Під категорією ресурсів «Ресурси Персональних Комп'ютерів» мається на увазі справна робота складових комп'ютерів, за допомогою яких оператори обробляють заяви абітурієнтів. В даному випадку, ресурс – будь який компонент ЕОМ та його можливості у виконанні поставлених задач. Ресурси можливо поділити на три складові: обчислювальні, файлові ,програмні. Схему такого розподілу зображено на рисунку 2.

Рисунок 2.2 – Схематичне зображення розподілу групи ресурсів «Ресурси ПК»

Кількість заяв, яку може обробити один персональній комп'ютер, що встановлений у відбірковій комісії може змінюватись в діапазоні від 50 до 75 в залежності від його характеристик.

2.1.3. Веб ресурси

Під веб-ресурсами мається на увазі посилання відповідні розділи приймальної комісії на сайті http://pk.kpi.ua/

2.2. Топологія

Топологією локальної мережі приймальної комісії КПІ ім. Ігоря Сікорського є зірка. Така топологія має наступну структуру:

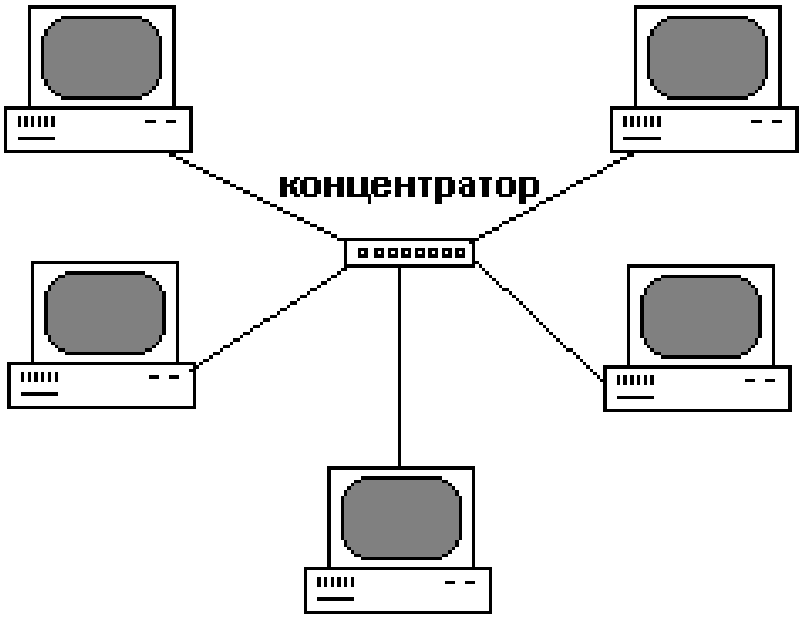

Рисунок 2.3 - Структура підключення при топології Зірка, що використовується в приймальній комісії.[14]

IP адреси призначаються статично (тобто за кожним комп'ютером закріплено конкретна IP адреса). Комп'ютери відбіркових комісій з'єднані з головним комп'ютером приймальної комісії в локальну мережу Ethernet кабелем.

Саме це з'єднання особливо актуальне для створення загального дискового простору, спільної роботи над документами.[1]

У мережі, побудованої за топологією типу «зірка», кожна робоча станція під'єднується кабелем (крученою парою) до концентратора, або хабу (англ. Hub). Концентратор забезпечує паралельне з'єднання ПК і, таким чином, всі комп'ютери, підключені до мережі, можуть спілкуватися один з одним.

Дані від передавальної станції мережі передаються через хаб по всіх лініях зв'язку всім ПК. Інформація надходить на всі робочі станції, але приймається тільки тими станціями, яким вона призначається. Так як передача сигналів в топології фізична зірка є широкомовної, тобто сигнали від ПК поширюються одночасно у всіх напрямках, то логічна топологія даної локальної мережі є логічною шиною.

Дана топологія застосовується в локальних мережах з архітектурою 10Base-T Ethernet.

Переваги мереж топології зірка:

1)легко підключити новий ПК;

2)є можливість централізованого управління;

4)мережа стійка до несправностей окремих ПК і до розривів з'єднання окремих ПК.

Серед недоліків мереж топології зірка можна виділити:

1)відмова хаба впливає на роботу всієї мережі;

2)велика витрата кабелю.

2.3. Висновки до розділу

В даному розділі було розглянуто специфіку роботи приймальної комісії КПІ ім.Ігоря Сікорського, показано можливий розподіл ресурсів відбіркових комісій факультетів\інститутів, розглянуто існуючі програмні додатки для контролю їх працездатності а також вказані їх переваги та недоліки. Розглянуто фізичну топологію локальної мережі.

3. АНАЛІЗ ІСНУЮЧИХ ПРОГРАМНИХ ЗАСОБІВ ДЛЯ ВИРІШЕННЯ ПОСТАВЛЕНОЇ ЗАДАЧІ, ЇХ ПЕРЕВАГИ ТА НЕДОЛІКИ

3.1. Cacti

Спочатку був MRTG - програмне забезпечення, що здійснює моніторинг мережі та вимірювання даних з плином часу. Автор програми, Тобіас Отікер (Tobias Oetiker) ще в 1990-х написав простий інструмент для побудови графіків, що використовує кільцеву базу даних, який на спершу використовувався для відображення пропускної здатності маршрутизатора в локальній мережі. Так MRTG породив RRDTool, набір утиліт для роботи з RRD, яка дозволяє зберігати, обробляти і графічно відображати динамічну інформацію, таку як температура , завантаження процесора, мережевий трафік і так далі. Нині RRDTool застосовують у величезній кількості інструментів з відкритим кодом. Сучасний флагман серед програмного забезпечення з відкритим вихідним кодом в області графічного представлення мережі –Cacti , виводить принципи MRTG на принципово новий рівень.

Безкоштовна програма, що входить в LAMP-набір серверного програмного забезпечення, яке надає стандартизовану програмну платформу для побудови графіків на основі практично будь-яких статистичних даних. Дані, швидше за все, можуть бути інтегровані в Cacti, якщо будь-який пристрій або сервіс повертає числові дані. Існують шаблони для моніторингу широкого спектру обладнання від Linux та Windows серверів до комутаторів і маршрутизаторів Cisco, — в основному все, що спілкується на SNMP (простий протокол мережевого управління). Існують також колекції шаблонів від сторонніх розробників, які ще більше розширюють і без того величезний список сумісних з Cacti апаратних засобів і програмного забезпечення.

Незважаючи на те, що стандартним методом збору даних Cacti є протокол SNMP, також для цього можуть бути використані сценарії Perl або PHP. Фреймворк програмної системи вміло поділяє на дискретні примірники збір даних та їх графічне відображення, що дозволяє з легкістю повторно обробляти і реорганізовувати існуючі

дані для різних візуальних уявлень. Крім того, можна обрати певні тимчасові рамки і окремі частини графіків просто натиснувши на них і перетягнувши. Так, наприклад, можна швидко переглянути дані за кілька минулих років, щоб зрозуміти, чи є поточна поведінка мережевого обладнання або сервера аномальним, або подібні показники з'являються регулярно. А використовуючи Network Weathermap, PHP-плагін для Cacti, ви без надмірних зусиль зможете створювати карти вашої мережі в реальному часі, що показують завантаженість каналів зв'язку між мережевими пристроями, які реалізуються з допомогою графіків, які з'являються при наведенні покажчика миші на зображення мережевого каналу. Багато організації, що використовують Cacti, виводять ці карти в цілодобовому режимі на 42-дюймові РК-монітори, встановлені на стіні, дозволяючи фахівцям миттєво відстежувати інформацію про завантаженість мережі і стан каналу.

Отже, Cacti - open-source веб-додаток, система дозволяє будувати графіки за допомогою RRDtool. Cacti збирає статистичні дані за певні часові інтервали і дозволяє відобразити їх у графічному вигляді. Переважно використовуються стандартні шаблони для відображення статистики по завантаженню процесора, виділенню оперативної пам'яті, кількістю запущених процесів, використання вхідного / вихідного трафіку.

3.2. Nagios

Nagios - програмна система для моніторингу мережі, яка вже багато років знаходиться в активній розробці. Написана на мові C, вона дозволяє робити майже все, що може знадобиться системним і мережевим адміністраторам від пакета прикладних програм для моніторингу. Веб-інтерфейс цієї програми є швидким та інтуїтивно зрозумілим, в той час його серверна частина - надзвичайно надійною.

Як і Cacti, спільнота підтримує Nagios, тому існують різні плагіни для величезної кількості апаратних засобів і програмного забезпечення. Від найпростіших ping-перевірок до інтеграції зі складними програмними рішеннями, такими як, наприклад, написаних на Perl безкоштовним програмним інструментарієм

WebInject для тестування веб-додатків і веб сервісів. Nagios дозволяє здійснювати постійний моніторинг стану серверів, мережевих каналів і всього іншого, що розуміє протокол мережевого рівня IP. Наприклад, можливо контролювати використання дискового простору на сервері, завантаженість ОЗУ і ЦП, використання ліцензії FLEXlm, температуру повітря на виході сервера, затримки в WAN та Інтернет-каналі і багато іншого.

Очевидно, що будь-яка система моніторингу серверів і мережі не буде повноцінною без повідомлень. У Nagios з цим все в порядку: програмна платформа пропонує налаштовувати механізм повідомлень по електронній пошті, через СМС ,миттєві повідомлення більшості популярних Інтернет-месенджерів, а також схему ескалації, яка може бути використана для прийняття рішень про те, хто, як і за яких обставин повинен бути повідомлений, що при правильному налаштуванні допоможе вам забезпечити багато годин спокійного сну. А веб - інтерфейс може бути використаний для тимчасового призупинення отримання сповіщень або підтвердження проблеми, і внесення правок адміністраторами.

Функція відображення також демонструє всі контрольовані пристрою в логічному поданні їх розміщення у мережі, з колірним кодуванням, це дозволяє виявити проблеми по мірі їх виникнення. Недоліком Nagios є конфігурація, через те що її краще всього виконувати через командний рядок, що значно ускладнює навчання новачків. Хоча люди, знайомі зі стандартними файлами конфігурації Linux/Unix, особливих проблем мати не повинні. Можливості Nagios величезні, але зусилля по використанню деяких з них не завжди актуальні та виправдані.

3.3. Icinga

Відгалуження від системи моніторингу Nagios - Icinga, нещодавно була представлена як самостійний проект, відомий як Icinga 2. Станом на сьогодні, обидві версії програми знаходяться в активній розробці і доступні для застосування, при цьому Icinga 1.x сумісна з великою кількістю плагінами і конфігурацією Nagios. Icinga 2 розроблялася менш громіздкою, з орієнтацією на продуктивність, і більш

зручною у використанні. Вона пропонує модульну архітектуру і багатий дизайн, яких немає ні в Nagios, ні в Icinga 1. Як і Nagios, Icinga може бути застосована для моніторингу всього, що говорить на мові IP, настільки глибоко, наскільки є можливим використовувати SNMP, а також налаштування, доповнення, плагіни.

Є кілька варіацій веб-інтерфейсу для Icinga, але головною особливістю цього програмного рішення для моніторингу від Nagios є конфігурація, яка може бути виконана через веб-інтерфейс, а не через файли конфігурації. Для тих, хто вважає за краще управляти своєю конфігурацією поза командного рядка, ця функціональність стане в нагоді.

Icinga інтегрується з багатьма програмними пакетами для моніторингу та графічного відображення, таких як PNP4Nagios, inGraph і Graphite, забезпечуючи надійну візуалізацію даної мережі. Крім того, Icinga має розширені можливості звітності.

3.4. NeDi

NeDi — це безкоштовне програмне забезпечення, що відноситься до LAMP, яка регулярно переглядає MAC-адреси та таблиці ARP в комутаторах мережі, збираючи кожен виявлений пристрій в локальній базі даних. Даний проект не є достатньо популярним, але він може стати дуже зручним інструментом при роботі з корпоративними мережами, де пристрої постійно змінюються і переміщаються. Існує також можливість через веб-інтерфейс NeDi задати пошук для визначення комутатора, до порту комутатора, точки доступу або будь-якого іншого пристрою по MAC-адресу, IP-адресу або DNS-імені. NeDi збирає всю інформацію, з кожного мережевого пристрою мережі, з якою стикається, витягаючи з них серійні номери, версії прошивки і програмного забезпечення, наявні параметри конфігурації модулів і т. д. Можливість використовувати NeDi для відзначення MAC-адрес пристроїв, які були втрачені. Якщо вони знову з'являться в мережі, NeDi повідомить про це.

Розпізнавання запускається процесом cron з заданими інтервалами. Конфігурація проста, з єдиним конфігураційним файлом, що надає можливість значно підвищити кількість налаштувань, в тому числі можливість пропускати пристрої на основі регулярних виразів або заданих меж мережі. NeDi, зазвичай, використовує протоколи Cisco Discovery Protocol або Link Layer Discovery Protocol для виявлення нових маршрутизаторів і комутаторів, а потім підключається до них для збору параметрів. Як тільки початкова конфігурація буде встановленою, виявлення пристроїв буде відбуватися швидко. До певного рівня NeDi може інтегруватися з Cacti, тому існує можливість зв'язати виявлення пристроїв з відповідними графіками Cacti.

3.5. Ntop

Ntop надає легко засвоювані графіки і таблиці, що показують поточний і минулий мережевий трафік, включаючи хости з обох кінців, джерело призначення та історію конкретних транзакцій, протокооли.

Також є можливість знайти набір графіків, діаграм і карт використання мережі в реальному часі, а також модульну архітектуру для величезної кількості надбудов, таких як додавання моніторів NetFlow і sFlow. Тут також є можливість знайти Nbox - апаратний монітор, який вбудовує в Ntop.

Крім того, Ntop включає API-інтерфейс для скриптової мови програмування Lua, який може бути використаний для підтримки розширень. Ntop також може зберігати дані хоста в файлах RRD для здійснення постійного збору даних. Одним з найбільш корисних застосувань Ntopng є контроль трафіку в конкретному місці. Наприклад, коли на карті мережі частина мережевих каналів підсвічено червоним, але незрозуміло, по якій причині користувач може за допомогою Ntopng отримати щохвилинний звіт про проблемний сегмент мережі та одразу дізнатися, які хости відповідальні за проблему. Користь від такої видимості мережі дуже велика. По суті, є можливість запустити Ntopng на будь-якому інтерфейсі, що був налаштований на рівні комутатора, для моніторингу іншого порту або VLAN.

3.6. Zabbix

Повномасштабний інструмент Zabbix для мережевого і системного моніторингу мережі, що об'єднує декілька функцій в одній веб-консолі. Забезпечуючи обслуговування і моніторинг продуктивності кожного об'єкта він може бути налаштований для моніторингу та збору даних з різних серверів і мережевих пристроїв.

В більшості, Zabbix працює з програмними агентами, запущеними на контрольованих системах. Таке рішення також може працювати і без агентів, використовуючи, наприклад, протокол SNMP. Надаючи детальні дані про продуктивність гіпервізора і його активності Zabbix підтримує VMware і інші гипервизоры віртуалізації. Особлива увага також приділяється моніторингу серверів додатків веб-сервісів і баз даних та Java. Хости можуть додаватися через процес автоматичного виявлення або вручну. Широкий набір шаблонів за замовчуванням застосовується до найбільш поширених варіантів використання, таких як Linux, FreeBSD і Windows-сервера; широко-служби, такі як SMTP, HTTP, а також ICMP і IPMI для детального моніторингу апаратної частини мережі. Користувальницькі перевірки, написані на Perl, Python або майже на будь-якому іншому мовою, можуть бути інтегровані в Zabbix. Щоб сфокусувати увагу на найбільш важливих компонентах мережі Zabbix дозволяє налаштовувати панелі моніторингу та веб-інтерфейс.

Попередження та ескалації проблем можуть ґрунтуватися на повторювальних діях, які застосовуються до групи хостів або хоста. Дії можуть навіть налаштовуватися для запуску віддалених команд, тому сценарій користувача може запускатися на контрольованому хості, якщо спостерігаються певні критерії подій. Програма відображає у вигляді графіків дані про продуктивність, такі як пропускна здатність мережі та завантаження процесора, а також збирає їх для власних систем відображення.

Крім того, Zabbix підтримує конфігурацію карти, екрани і навіть слайд-шоу, що відображають поточний статус контрольованих пристроїв. Zabbix може бути складним для реалізації на початковому етапі, але розумне використання автоматичного виявлення і різних шаблонів може частково полегшити труднощі з інтеграцією. На додаток до встановлюваному пакету, Zabbix доступний як віртуальний додаток для відомих гіпервізорів.

3.7. Observium

Програмне забезпечення для моніторингу серверів та мережевого обладнання - Observium, має різноманітний список підтримуваних пристроїв, що використовують протокол SNMP. Відносячись до LAMP, Observium відносно легко встановлюється і налаштовується, вимагаючи звичайних установок Apache, PHP і MySQL, створення бази даних, конфігурації Apache і тому подібного. Програмний додаток встановлюється як власний сервер з виділеною URL-адресою. Є можливість увійти в графічний інтерфейс і почати додавати хости і мережі, а також задати діапазони для автоматичного виявлення і дані SNMP, щоб Observium міг досліджувати навколишні його мережі і збирати дані по кожній виявленій системі.

Щоб допомогти в зборі даних Observium також має можливість виявляти мережеві пристрої через протоколи CDP, LLDP або FDP, а віддалені агенти хоста можуть бути розгорнуті на Linux-системах. Всі зібрані дані доступні через інтерфейс, який надає широкі можливості для статистичного відображення даних, а також у вигляді графіків і діаграм. Програма надає можливість отримати час відгуку ping і SNMP , графіки пропускної здатності, фрагментації, кількості IP-пакетів і багато іншого. Ці дані можуть бути доступні аж для кожного виявленого порту,в залежності від пристрою.

Що до серверів, то для них Observium може відобразити інформацію про стан центрального процесора, свопу, температури оперативної пам'яті, сховища даних і т.п. Існує функція збору даних і графічного відображення продуктивності для різних сервісів, включаючи Apache, MySQL, BIND, Memcached, Postfix і інші. Програма

може швидко стати основним інструментом для отримання інформації про стан серверів і мереж працювати як віртуальна машина. Це відмінний спосіб додати графічне представлення в мережу будь-якого розміру а також автоматичне виявлення.

3.8. Висновки до розділу

В даному розділі було проаналізовано існуючі програмні засоби для вирішення поставленої задачі, їх переваги та недоліки

4. ЗАСОБИ РЕАЛІЗАЦІЇ ПРОГРАМНОЇ СИСТЕМИ

4.1. Мова програмування Python

Python представляє собою інтерпретовану об'єктно-орієнтовану мову і інтерактивне середовище для розробки програм. З його допомогою можна розробляти програми з графічним інтерфейсом, працювати з базами даних, створювати Web-сайти. Мова програмування Python має досить зрозумілий синтаксис і є зручною для програмування математичних обчислень.

Основні фактори, які можна відмітити, підкреслюючи акуальність і зручність використання цієї мови програмування [4]:

1) Якість програмного забезпечення

Для багатьох основна перевага мови Python полягає в легкості читання, ясності і більш високій якості, що відрізняють його від інших інструментів в світі мов сценаріїв. Програмний код на мові Python читається легше, а значить, багаторазове його використання і обслуговування виконується набагато простіше, ніж використання програмного коду на інших мовах сценаріїв. Одноманітність оформлення програмного коду на мові Python полегшує його розуміння навіть для тих, хто не брав участі в його створенні. Крім того, Python підтримує найсучасніші механізми багаторазового використання програмного коду, яким є об'єктно-орієнтоване програмування (ООП).

2) Висока швидкість розробки

У порівнянні з мовами, що компілюються або є (C, C ++ і Java), Python у багато разів підвищує продуктивність праці розробника. Обсяг програмного коду на мові Python становить третину або навіть п'яту частину еквівалентного програмного коду на мові C ++ або Java. Це означає менший обсяг введення з клавіатури, менша

кількість часу на налагодження і менший обсяг трудовитрат на супровід. Крім того, програми на мові Python запускаються відразу ж, минаючи тривалі етапи компіляції і зв'язування, які необхідні в деяких інших мовах програмування.

3) Кросплатформлена мова програмування

Велика частина програм на мові Python виконується без змін на всіх основних платформах. Перенесення програмного коду з операційної системи Linux в Windows зазвичай полягає в простому копіюванні файлів програм з однієї машини на іншу. Більш того, Python надає масу можливостей по створенню переносних графічних інтерфейсів, програм доступу до баз даних, веб-додатків і багатьох інших типів програм. Навіть інтерфейси операційних систем, включаючи спосіб запуску програм і обробку каталогів, в мові Python реалізовані переносимим способом.

3) Бібліотеки

У складі Python наявна велика кількість зібраних і переносних функціональних можливостей, відомих як стандартна бібліотека. Оскільки мова програмування є безкоштовною, то майже всі нові розробки бібліотек наявні в онлайн доступі. Робота з великими базами даних дає можливість опрацювання статистичних задач, що досить часто виникають при рішенні основної задачі. Ця бібліотека надає масу можливостей, що є актуальними в прикладних програмах, починаючи від пошуку тексту по шаблону і закінчуючи мережевими функціями. Крім того, Python допускає розширення як за рахунок власних бібліотек, так і за рахунок бібліотек, створених сторонніми розробниками. З числа сторонніх розробок можна назвати інструменти створення веб-сайтів, програмування математичних обчислень, доступ до послідовного порту, розробку ігрових програм і багато іншого.

4) Інтеграція компонентів

Сценарії Python легко можуть взаємодіяти з іншими частинами додатку завдяки різним механізмам інтеграції. Ця інтеграція дозволяє використовувати Python для настройки і розширення функціональних можливостей програмних продуктів. На

сьогоднішній день програмний код на мові Python має можливість викликати функції з бібліотек на мові C / C ++, сам викликатися з програм, написаних на мові C / C ++, інтегруватися з програмними компонентами на мові Java, взаємодіяти з такими платформами, як COM і .NET , і проводити обмін даними через послідовний порт або через мережу за допомогою таких протоколів, як SOAP, XML-RPC і CORBA. Python - не відокремлений інструмент.

Python - інтерпретована, об'єктно-орієнтована мова програмування високого рівня з динамічною семантикою. Високорівневі вбудовані структури даних у поєднанні з динамічним набором тексту та динамічним зв'язуванням роблять її дуже актуальною для швидкого розвитку додатків, а також для використання в якості мови сценаріїв або клею для з'єднання існуючих компонентів разом.

4.2. Інтегроване середовище розробки PyCharm

PyCharm – інтегроване середовище розробки для мови програмування Python. Надає засоби для аналізу коду, графічний відладчик, інструмент для запуску юніт-тестів і підтримує веб-розробку на Django. PyCharm розроблена компанією JetBrains на основі IntelliJ IDEA [6,7].

Середовище розробки володіє наступними властивостями:

- Статичний аналіз коду, підсвічування синтаксису і помилок.
- Навігація по проекту і вихідного коду: відображення файлової структури проекту, швидкий перехід між файлами, класами, методами і використаннями методів.
- Рефакторинг: перейменування, вилучення методу, введення змінної, введення константи, підйом і спуск методу і т. д.
- Інструменти для веб-розробки з використанням фреймворку Django
- Вбудований відладчик для Python
- Вбудовані інструменти для юніт-тестування
- Розробка з використанням Google App Engine

-Підтримка систем контролю версій: загальний користувальницький інтерфейс для Mercurial, Git, Subversion, Perforce і CVS з підтримкою списків змін.

4.3 Висновки до розділу

Порівнюючи поставлені задачі та провівши огляд можливостей мов програмування було обрано Python, як оптимальну мову для адміністраторів. До переваг даної мови програмування, можна віднести можливість:

- адміністрування;
- автоматизації процесів;
- моніторингу процесів та систем;
- embedded системи.

А безкоштовність мови програмування забезпечує майже неперервне поповнення бібліотек в онлайн доступі.

До недоліків можна віднести:

- динамічну типізацію даних, внаслідок якої запуск програми та оброблення нею інформації є повільнішим від аналогів;
- відсутній функціонал для створення звітів.

Дані недоліки не є принциповими , а от переваги достатньо переконливі щоб зупинити вибір на мові програмування Python.

5. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

5.1. Опис роботи програмного додатку, логічна структура

Програмний додаток визначає кількість людських ресурсів за конкретну дату порівнюючи скільки заяв абітурієнтів повинна обробити відбіркова комісія кожного факультету і скільки спроможна. Для визначення кількості документів, які необхідно обробити, програмний додаток зчитує Excel –файл, в якому вказана дата, час подачі заяви абітурієнта до ЄДЕБО і назва факультету.

	Структурний підрозділ	Час додання заяви до ЄДЕБО
2	Теплоенергетичний факультет	11.08.2018 14:02
3	Теплоенергетичний факультет	11.08.2018 14:00
4	Видавничо-поліграфічний інститут	08.08.2018 17:44
5	Механіко-машинобудівний інститут	08.08.2018 17:35
6	Факультет менеджменту та маркетингу	08.08.2018 16:23
7	Факультет менеджменту та маркетингу	08.08.2018 16:22
8	Факультет менеджменту та маркетингу	08.08.2018 16:21
9	Факультет лінгвістики	04.08.2018 09:37
10	Радіотехнічний факультет	03.08.2018 19:00
11	Інститут телекомунікаційних систем	03.08.2018 18:20
12	Факультет соціології і права	03.08.2018 18:19
13	Факультет менеджменту та маркетингу	03.08.2018 18:01
14	Факультет менеджменту та маркетингу	03.08.2018 18:01
15	Факультет інформатики та обчислювальної техніки	03.08.2018 17:53
16	Факультет менеджменту та маркетингу	03.08.2018 17:50
17	Факультет інформатики та обчислювальної техніки	03.08.2018 17:50
18	Факультет менеджменту та маркетингу	03.08.2018 17:49
19	Факультет менеджменту та маркетингу	03.08.2018 17:45
20	Факультет соціології і права	03.08.2018 17:44
21	Факультет соціології і права	03.08.2018 17:40

Рисунок 5.1 - Структура Excel-файлу

Для визначення кількості документів, які спроможна обробити відбіркова комісія кожного факультету, програмний додаток відкриває і зчитує файл з даними про параметри кожного ПК відбіркової комісії і аналізує їх. Для зчитування даних з віддаленого комп’ютера застосовується програма- агент

Після отримання даних про параметри комп’ютера програма аналізує їх наступним чином:

В залежності від виробника відеокарти:

Таблиця 5.1 – Залежність кількості заяв від виробника відеокарти

Виробник відеокарти	К-кість заяв,що віднімаються від максимальної кількості(75)
3D Labs	1
Matrox	1
ASRock	1
PNY Technologies	1
SiS	1
Abit	2
Palit	2
Sapphire	2
Trident Microsystems	2
інші	0

В залежності від операційної системи:

Таблиця 5.2 – Залежність кількості заяв від операційної системи

Система	К-кість заяв,що віднімаються від максимальної кількості(75)
Windows 7.0	1
Windows міленіум	4
Windows xp	4
Windows 97	5
Windows 95	6
інші	0

В залежності від тактової частоти процесора:

Таблиця 5.3 – Залежність кількості заяв від тактової частоти процесора

Тактова частота процесора	К-кість заяв,що віднімаються від максимальної кількості(75)
0 -12,5	5
12,5 - 66	3
66 - 100	2
100 -150	1
вище	0

В залежності від процесора:

Таблиця 5.4 – Залежність кількості заяв від процесора

Процесори	К-кість заяв,що віднімаються від максимальної кількості(75)
Intel Core i3-6100	1
Intel Core i3-6300	
Intel Core i3-6320	
Intel Core i3-7100	
Intel Core i3-7300	
Intel Core i3-7350K	
Intel Core i3-8100	
Intel Core i3-8100T	
Intel Core i3-8300	
Intel Core i3-8300K	
Intel Core i3-9100	
Intel Core i3-9100F	
Intel Core i3-9300	
Intel Core i5-4460	
Intel Core i5-4590	
Intel Core i5-6500	2
Intel Core i5-6600	
Intel Core i5-6600K	
Intel Core i5-7400	
Intel Core i5-7500	
Intel Core i5-7600	
Intel Core i5-7600K	
Intel Core i5-7640X	
Intel Core i5-8400	
Intel Core i5-8500	
Intel Core i5-8600	
Intel Core i5-8600K	
Intel Core i5-9400	
Intel Core i5-9400F	
інші	0

В залежності від всього об’єму фізичної пам’яті:

Таблиця 5.5 – Залежність кількості заяв від всього об’єму фізичної пам’яті

Максимальний об'єм фізичної пам'яті	К-кість заяв,що віднімаються від максимальної кількості(75)
0 Гб-8 Гб	15
8 Гб -16 Гб	10
16 Гб - 32 Гб	8
32 Гб - 64 Гб	5
64 Гб - 128 Гб	3
128 Гб -254 Гб	2
254 Гб - 512 Гб	1

В залежності від доступного об’єму фізичної пам’яті :

Таблиця 5.6 – Залежність кількості заяв від доступного об’єму фізичної пам’яті

Максимальний об'єм фізичної пам'яті	К-кість заяв,що віднімаються від максимальної кількості(75)
0 Гб-8 Гб	15
8 Гб -16 Гб	10
16 Гб - 32 Гб	8
32 Гб - 64 Гб	5
64 Гб - 128 Гб	3
128 Гб -254 Гб	2
254 Гб - 512 Гб	1

Після отримання і аналізу необхідних даних, програма, обчислює необхідні людські ресурси, враховуючи, що на 1 працюючий ПК необхідний 1 технічний секретар,1 оператор та 1 відповідальний за факультет/інститут

Логічна структура ПД UML-діаграма

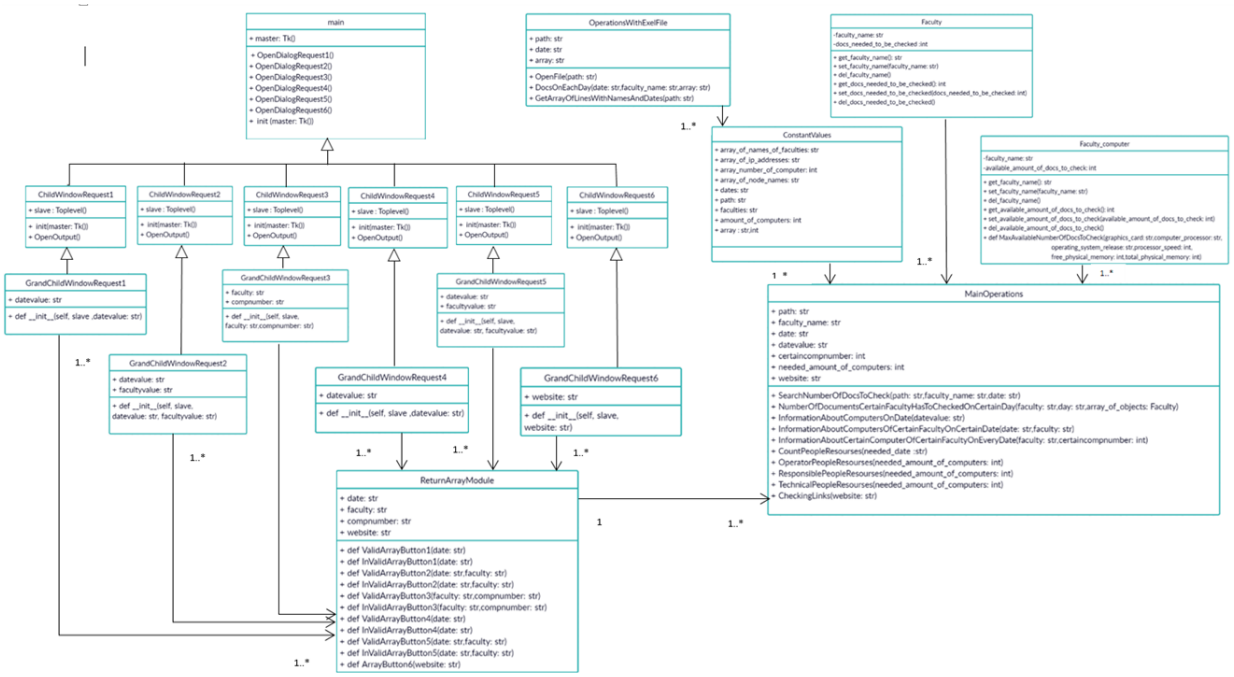

Рисунок 5.2 - Логічна структура ПД UML-діаграма

5.2. Опис модулів/бібліотек

5.2.1. Модуль datetime

Модуль datetime входить в стандартну бібліотеку мови програмування Python і представляє собою збірник з широкого спектру класів для комфортної роботи з датами і часом. Функціональність програм підвищується за рахунок безлічі вбудованих методів, призначених для зручного відображення, а також маніпуляції над часом і датами.

Бібліотека datetime використовується для роботи в Python з часом і датами, дозволяючи представляти дану інформацію в найбільш зручній формі. Вона складається з декількох класів. Завдяки їх наявності, програміст може отримати доступ до таких методів:

- Операції, які дозволяють порівнювати час;
- Форматований вивід інформації про дату і часі

- Отримання поточних системних дати і часу;
- Обчислення різниці між датами та інші арифметичні операції;

Кожен з класів модуля володіє власними методами і властивостями, а також служить для певних цілей.

5.2.2. Модуль tkinter

Пакет для Python - tkinter, передбачений для роботи з бібліотекою Tk. Бібліотека Tk містить компоненти графічного інтерфейсу користувача (графічний інтерфейс користувача - GUI), написані на мовній програмі програми Tcl. .[11]

Під графічним інтерфейсом користувача (GUI) підключаються всі теки, текстові поля для вводу, скроллери, радіокнопки, флажки кнопки,вікна,списки. За їх допомогою можна взаємодіяти з програмою і керувати нею. Елементи інтерфейсу разом з ними будуть називатись віджетами

Графічний інтерфейс нині мають майже всі програми, які створюються для конкретного користувача. Існують різні бібліотеки GUI. З використанням Tk написано не мало проектів, але це не розповсюджене рішення . Хоча з ряду причин вона була вибрана для Python за замовчуванням. Зазвичай Python вже включає пакет tkinter у складі стандартних бібліотек, що містяться з іншими модулями.

Tkinter виступає як перекладач з мови Python на мову Tcl. Ви пишете програму на Python, і код модуля tkinter у вас для спільного переказу ваші інструкції на мові Tcl, який позначає бібліотеку Tk.

5.2.3. Модуль socket

Цей модуль забезпечує доступ до інтерфейсу BSD. Він доступний на всіх сучасних системах MacOS , Unix, Windows в тому числі на додаткових платформах.

5.2.4. Модуль os

Модуль os надає можливість взаємодіяти з операційною системою - дізнаватися або змінювати файлову структуру, дізнаватися ім'я і права користувача, змінні середовища і ін. Оскільки os вміє враховувати особливості кожної ОС, програма, що використовує змінні і функції модуля os, переносима з однієї операційної системи на іншу. Однак ряд функцій використовується тільки для Unix-подібних ОС.

Інтерфейс Python - це проста транслітерація системного інтерфейсу виклику та бібліотеки Unix для сокетів до об'єктно-орієнтованого стилю Python: функція socket () повертає об'єкт сокета, методи якого реальзують різні системні виклики сокет. Типи параметрів дещо вищого рівня, ніж у інтерфейсі C: як і при операціях читання () і запису () на файлах Python, розподіл буфера в операціях прийому є автоматичним, а довжина буфера неявна для операцій надсилання.

5.2.5. Модуль xlrd

Модуль експортує дані з електронних таблиць Excel (.xls та .xlsx, версії 2.0) на будь-якій платформі. Підтримує версії python (2,7, 3,4+). Наявна можливість підтримки дат Excel. Unicode- сумісний. .[12]

5.2.6. Модуль wmi

Інструмент управління Windows (WMI) - це впровадження Microsoft Web-based Enterprise Management (WBEM), галузева ініціатива щодо створення загальної інформаційної моделі (CIM) для майже будь-якої інформації про комп'ютерну систему.

5.2.7. Модуль bs4

Beautiful Soup - бібліотека Python для витягу даних з HTML та XML-файлів. Вона розташована на вершині HTML або XML-аналізатора, надаючи пітонічні ідіоми для ітерації, пошуку та модифікації дерева розбору.

5.2.8. Модуль psutil

psutil (Python System і утиліти процесів) - це кроссплатформена бібліотека для отримання інформації про запущені процеси і використанні системи (CPU, пам'ять, диски, мережа) в Python. Це корисно в основному для моніторингу системи, профілювання і обмеження ресурсів процесів і управління запущеними процесами. Він реалізує безліч функцій, пропонованих інструментами командного рядка, такими як: ps, top, lsof, netstat, ifconfig, який, df, kill, free, nice, ionice, iostat, iotop, uptime, pidof, tty, taskset, pmap. В даний час він підтримує Linux, Windows, OSX, Sun Solaris, FreeBSD, OpenBSD та NetBSD, як 32-розрядні, так і 64-розрядні архітектури, з версіями Python від 2.6 до 3.5 (користувачі Python 2.4 і 2.5 можуть використовувати версію 2.1. 3).

5.2.9. Модуль Openpyxl

openpyxl - це бібліотека Python для читання / запису файлів Excel 2010 xlsx / xlsm / xltx / xltm.

Вона народилася через відсутність існуючої бібліотеки для читання / запису з Python, формату Office Open XML.

5.2.10. Модуль Numpy

Numpy - це універсальний пакет для обробки масивів. Він надає високопродуктивний об'єкт багатовимірного масиву і інструменти для роботи з цими масивами. Це фундаментальний пакет для наукових обчислень з Python.

Крім очевидного наукового використання, Numpy також може використовуватися в якості ефективного багатовимірного контейнера загальних даних.

Масив в Numpy є таблицею елементів (зазвичай чисел) одного типу, проіндексованих набором натуральних чисел. У Numpy число вимірювань масиву називається рангом масиву. Кортеж цілих чисел, що визначає розмір масиву по кожному виміру, називається формою масиву. Клас масиву в Numpy називається ndarray. Елементи в масивах Numpy доступні за допомогою квадратних дужок і можуть бути ініційовані за допомогою вкладених списків Python.

5.2.11. Модуль Itertools

Модуль Itertools реалізує ряд будівельних блоків ітераторів, заснованих на конструкціях з APL, Haskell і SML. Кожен був перетворений в форму, придатну для Python.

Модуль стандартизує основний набір швидких, ефективних по пам'яті інструментів, які корисні самі по собі або в поєднанні. Разом вони утворюють «алгебру ітераторів», що дозволяє швидко і ефективно створювати спеціалізовані інструменти в Python.

5.2.12. Модуль Platform

Модуль Platform використовується для отримання максимально можливої інформації про платформу, на якій в даний момент виконується програма. Тепер під інформацією про платформу мається на увазі інформація про пристрій, його ОС,

вузлі, версії ОС, версії Python і т. Д.Цей модуль грає важливу роль, коли ви хочете перевірити, чи сумісна Ваша програма з версією python, встановлена в конкретній системі, або чи відповідають специфікації обладнання вимогам користувацької програми.

Цей модуль вже існує в бібліотеці Python і не вимагає установки з використанням pip.

5.2.13. Модуль Schedule

Модуль Schedule - це внутріпроцессний планувальник для періодичних завдань, які використовують шаблон будівника для конфігурації. Розклад дозволяє періодично запускати функції Python (або будь-які інші викликаються) через заздалегідь певні інтервали, використовуючи простий, зрозумілий людині синтаксис.

Бібліотека розкладу використовується для планування завдання в певний час кожен день або в певний день тижня. Ми також можемо встановити час в 24-годинному форматі, щоб під час запуску завдання. По суті, бібліотека розкладів зіставляє системне час із запланованим вами часом. Коли запланований час і системний час збігаються, викликається функція завдання (запланована командна функція).

5.2.14. Модуль Time

Модуль time - це вбудований модуль в Python, і він має різні функції, які вимагають виконання більшої кількості операцій вчасно. Це один з кращих модулів в Python, який використовується для вирішення різних реальних проблем, пов'язаних з часом. Щоб використовувати модуль часу в програмі, спочатку необхідно імпортувати модуль часу.

5.2.15. Модуль requests

Модуль requests дозволяє відправляти HTTP-запити з використанням Python.

HTTP-requests повертає об'єкт відповіді з усіма даними відповіді (вміст, кодування, стан і т. д.).

Запити можна надсилати запити HTTP / 1.1 надзвичайно легко. Немає необхідності вручну додавати рядки запиту в URL-адреси або кодувати дані POST у формі. Keep-alive та пул HTTP-з'єднань на 100% автоматично, завдяки urllib3.

5.2.16. Модуль urllib.parse

Модуль дає можливість визначити стандартний інтерфейс для розбиття рядків уніфікованого локатора ресурсів (URL) на компоненти (схема адресації, розташування мережі, шлях тощо), об'єднання компонентів назад у рядок URL та перетворення "відносної URL-адреси" з "базової URL-адреси" в абсолютну URL-адресу. [13]

Модуль розроблений таким чином, щоб відповідати Інтернет-RFC на відносних уніфікованих локаторах ресурсів. Він підтримує такі схеми URL: файл , prospero, rsync, rtsp, rtspu, sftp, shttp, sip, sips, snews, svn, svn + ssh, telnet, wais, ws, wss. Модуль urllib.parse визначає функції, які поділяються на дві широкі категорії: розбір URL-адрес та цитування URL-адрес. Вони детально розглядаються в наступних розділах. Розбірка URL- адреси . Функції розбору URL-адрес фокусуються на розділенні рядка URL-адреси на його компоненти або на об'єднанні компонентів URL-адреси в рядку URL-адреси.

5.3. Створення і опис роботи класів

5.3.1. Клас Faculty_computer

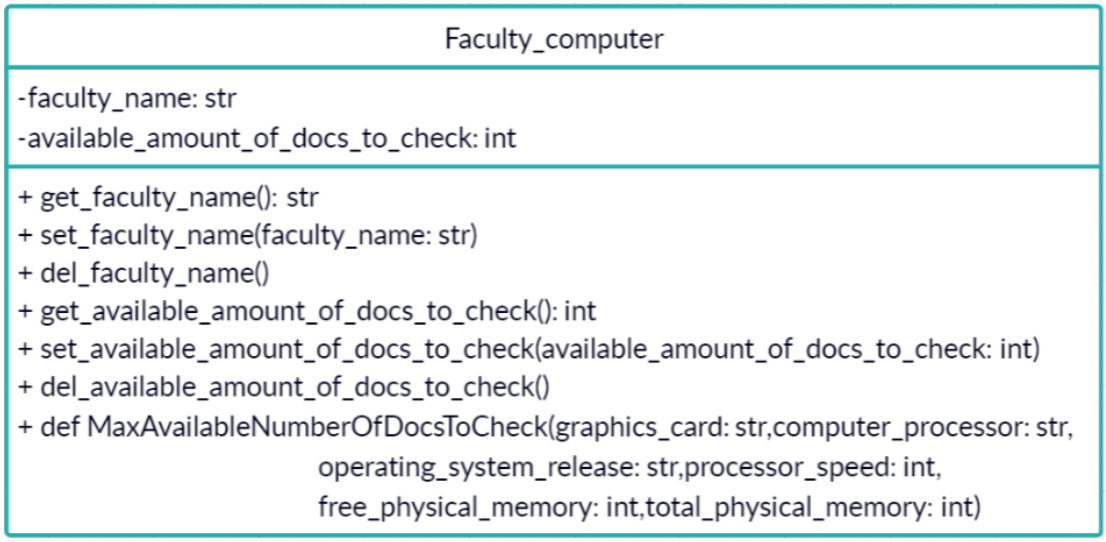

Рисунок 5.3 - Діаграма класу Faculty_computer

Даний клас описує кожний ПК відбіркових комісій факультетів/інститутів

Таблиця 5.7 – Опис полів класу Faculty_computer:

№ з/п	Назва	Тип	Призначення
1	faculty_name	Str	Для зберігання значення назви факультету/інституту
2	number_of_computer	Int	Для зберігання значення порядкового номеру комп'ютера
3	computer_ip_address	Str	Для зберігання значення ip адреси комп'ютера
4	node_name	Str	Для зберігання значення назви вузла
5	os_name	Str	Для зберігання значення назви операційної системи
6	os_release	Str	Для зберігання значення

7	Manufacturer	Str	Для зберігання значення назви виробника
8	proceccor_name	Str	Для зберігання значення назви пооцесору
9	proceccor_speed	Int	Для зберігання значення швидкості процесору
10	free_physical_memory	Int	Для зберігання значення доступної памяті
11	total_physical_memory	Int	Для зберігання значення повного обєму фізичної памяті
12	percent_of_used_memory	Int	Для зберігання значення відсотка використаної памяті
13	available_amount_of_docs_to_check	Int	Для зберігання значення максимально можливого числа документів на перевірку
14	Date	Int	Для зберігання значення дати

Таблиця 5.8 – Опис методів класу Faculty_computer:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self,faculty_name, number_of_compuer, computer_ip_address, node_name, os_name, os_release, manufacturer, proceccor_name,date, proceccor_speed, free_physical_memory, total_physical_memory, percent_of_used_memory, available_amount_of_docs_to_check)	self,faculty_name, number_of_compuer, computer_ip_address, node_name, os_name, os_release, manufacturer, proceccor_name,date, proceccor_speed, free_physical_memory, total_physical_memory, percent_of_used_memory,available_amount_of_docs_to_check)	–	Конструктор за замовчуванням
2	def get_faculty_name(self)	-	faculty_name	Функція для повернення значення значення назви факультету/інституту
3	def set_faculty_name(self, faculty_name)	faculty_name	-	Функція для встановлення

				значення назви факультету/інституту
4	def del_faculty_name(self)	-	-	Деструктор
5	def get_number_of_compuer(self)	-	number_of_compuer	Функція повернення значення порядкового номеру комп'ютера
6	def set_number_of_compuer(self, number_of_compuer)	number_of_compuer	-	Функція для встановлення значення порядкового номеру комп'ютера
7	def del_number_of_compuer(self)	-	-	Деструктор
8	def get_computer_ip_address(self)	-	computer_ip_address	Функція для повернення значення Ip-адреси
9	def set_computer_ip_address(self, computer_ip_address)	computer_ip_address	-	Функція для встановлення значення Ip-адреси
10	def del_computer_ip_address(self)	-	-	Деструктор
11	def get_node_name(self)	-	node_name	Функція для повернення значення імені вузла
12	def set_node_name(self,node_name)	node_name	-	Функція для встановлення значення назви імені вузла
13	def del_node_name(self)	-	-	Деструктор
14	def get_os_name(self)	-	os_name	Функція для повернення значення операційної системи
15	def set_os_name(self,os_name)	os_name	-	Функція для встановлення значення назви операційної системи
16	def del_os_name(self)	-	-	Деструктор
17	def get_os_release(self)	-	os_release	Функція для повернення значення релізу операційної системи
18	def set_os_release(self,os_release)	os_release	-	Функція для встановлення значення релізу операційної системи
19	def del_os_release(self)	-	-	Деструктор
20	def get_manufacturer(self)	-	Manufacturer	Функція для повернення значення назви виробника відеокарти

21	def set_manufacturer(self,manufacturer)	Manufacturer	-	Функція для встановлення значення назви виробника відеокарти
22	def del_manufacturer(self)	-	-	Деструктор
23	def get_proceccor_name(self)	-	proceccor_name	Функція для повернення значення назви процесора
24	def set_proceccor_name(self,proceccor_name)	proceccor_name	-	Функція для встановлення значення назви процесора
25	def del_proceccor_name(self)	-	-	Деструктор
26	def get_proceccor_speed(self)	-	proceccor_speed	Функція для повернення значення тактової частоти процесору
27	def set_proceccor_speed(self,proceccor_speed)	proceccor_speed	-	Функція для встановлення значення тактової частоти процесору
28	def del_proceccor_speed(self)	-	-	Деструктор
29	def get_free_physical_memory(self)	-	free_physical_memory	Функція для повернення значення доступної фізичної пам’яті
30	def set_free_physical_memory(self,free_physical_memory)	free_physical_memory	-	Функція для встановлення значення доступної фізичної пам’яті
31	def del_free_physical_memory(self)	-	-	Деструктор
32	def get_total_physical_memory(self)	-	total_physical_memory	Функція для повернення значення загального об’єму фізичної пам’яті
33	def set_total_physical_memory(self,total_physical_memory)	total_physical_memory	-	Функція для встановлення значення загального об’єму фізичної пам’яті
34	def del_total_physical_memory(self)	-	-	Деструктор
35	def get_percent_of_used_memory(self)	-	percent_of_used_memory	Функція для повернення значення Відсотка використаної фізичної пам’яті
36	def set_percent_of_used_memo	percent_of_used_memory	-	Функція для встановлення значення

	ry(self,percent_of_used_memory)			відстотка використаної фізичної пам’яті
37	def del_percent_of_used_memory(self)	-	-	Деструктор
38	def get_available_amount_of_docs_to_check(self):	-	available_amount_of_docs_to_check	Функція для повернення значення available_amount_of_docs_to_check
39	def set_available_amount_of_docs_to_check(self,available_amount_of_docs_to_check)	available_amount_of_docs_to_check	-	Функція для встановлення значення available_amount_of_docs_to_check
40	def del_available_amount_of_docs_to_check(self)	-	-	Деструктор
41	def get_date(self):	-	self.__date	Функція для повернення значення Дати
42	def set_date(self,date)	Date	-	Функція для встановлення значення дати
43	def del_date(self):	-	-	Деструктор
44	def MaxAvailableNumberOfDocsToCheck(graphics_card,computer_processor, operating_system_release,processor_speed, free_physical_memory,total_physical_memory)	graphics_card,computer_processor, operating_system_release,processor_speed, free_physical_memory, total_physical_memory	-	Функція для визначення максимального числа заяв, які спроможний обробити комп’ютер

5.3.2. Клас Faculty

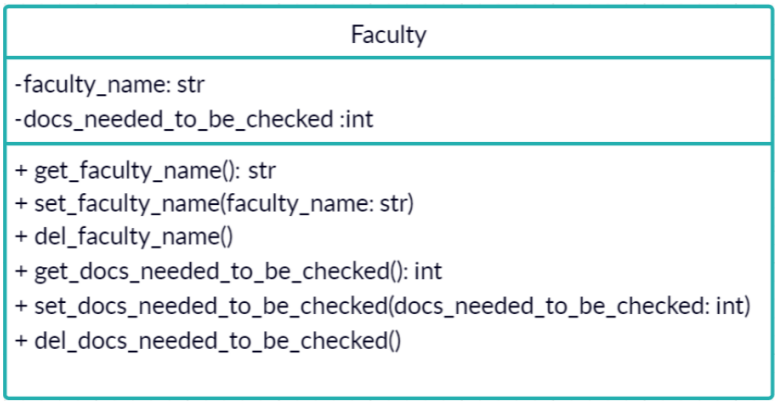

Рисунок 5.4 - Діаграма класу Faculty

Даний клас описує факультет/інститут НТУУ «КПІ ім. Ігоря Сікорського»

Таблиця 5.9 – Опис полів класу Faculty:

№ з/п	Назва	Тип	Призначення
1	faculty_name	Str	Для зберігання назви факультету/інституту
2	amount_of_computers	Int	Для зберігання значення порядкового номеру комп’ютера факультету/інституту
3	Date	Str	Для зберігання значення дати
4	docs_needed_to_be_checked	Int	Для зберігання чисельного значення документів, які необхідно перевірити

Таблиця 5.8 – Опис методів класу Faculty:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__ (self,faculty_name, amount_of_computers,date,docs_needed_to_be_checked)	self,faculty_name,amount_of_computers,date,docs_needed_to_be_checked	-	Конструктор за замовчуванням
2	def get_faculty_name(self)	-	faculty_name	Функція для повернення значення назви факультету
3	def set_faculty_name(self, faculty_name)	faculty_name	-	Функція для встановлення значення назви факультету
4	def del_faculty_name(self)	-	-	Деструктор
5	def get_amount_of_computers(self)	-	amount_of_computers	Функція для повернення значення кількості комп’ютерів
6	def set_amount_of_computers(self, amount_of_computers)	amount_of_computers	-	Функція для встановлення значення кількості комп’ютерів

7	def del_amount_of_computers(self)	-	-	Деструктор
8	def get_date(self)	-	Date	Функція для повернення значення дати
9	def set_date(self, date)	Date	-	Функція для встановлення значення дати
10	def del_date(self)	-	-	Деструктор
11	def get_docs_needed_to_be_checked(self)	-	docs_needed_to_be_checked	Функція для повернення значення числа документів, які необхідно перевірити
12	def set_docs_needed_to_be_checked(self, docs_needed_to_be_checked)	docs_needed_to_be_checked	-	Функція для встановлення значення числа документів, які необхідно перевірити
13	def del_docs_needed_to_be_checked(self)	-	-	Деструктор

5.3.3. Клас ConstantValues

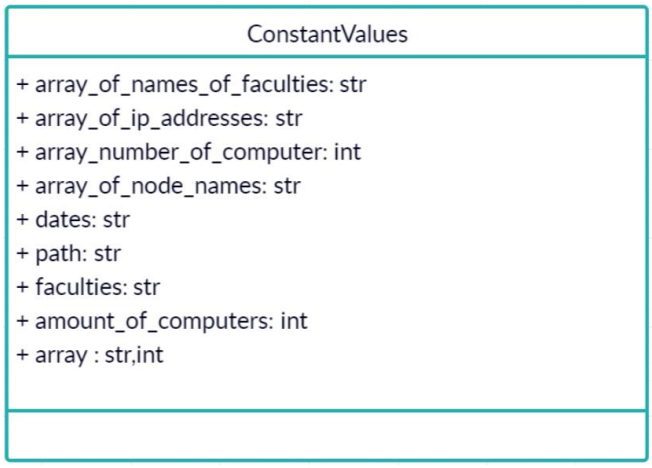

Рисунок 5.5 - Діаграма класу ConstantValues

Клас ConstantValues призначений для зберігання змінних, які будуть забезпечувати функціонал і не змінюватимуть свого значення впродовж всього часу виконання програми

5.3.4. Клас OperationsWithExcelFile

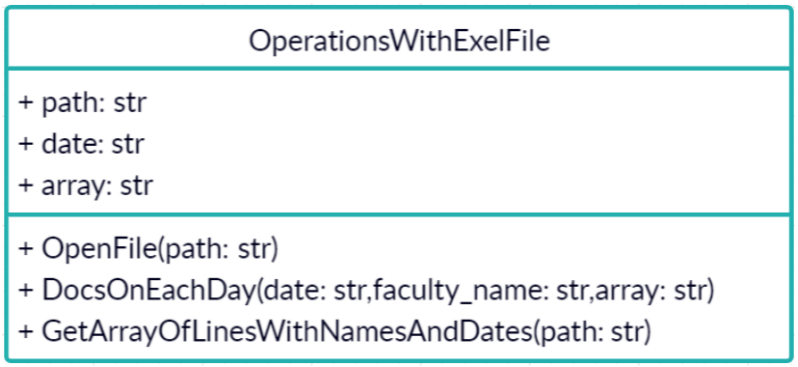

Рисунок 5.6 - Діаграма класу OperationsWithExcelFile

Класс OperationsWithExcelFile створений для роботи з вхідним Excel–файлом, зчитування та аналізу, отриманої з нього інформації

Таблиця 5.9 – Опис методів класу OperationsWithExcelFile:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def OpenFile(path)	Path - шлях до файлу	array_of_lines_with_names_and_dates – масив з назвою факультету та датою подачі заяви	Одержання масиву назвою факультету та датою подачі заяви з вказаного файлу
2	def DocsOnEachDay(date,faculty_name,array)	date, faculty_name, array	Counter – к-кість заяв,які необхідно обробити за кожний день	Одержаня кількості заяв,які необхідно обробити за кожний день

3	def GetArrayOfLinesWithNamesAndDates(path)	Path - шлях до файлу	array_of_all_dates_and_faculties	Одержання масиву з назвами факультетів, датами, кількістю заяв

5.3.5. Клас MainOperations

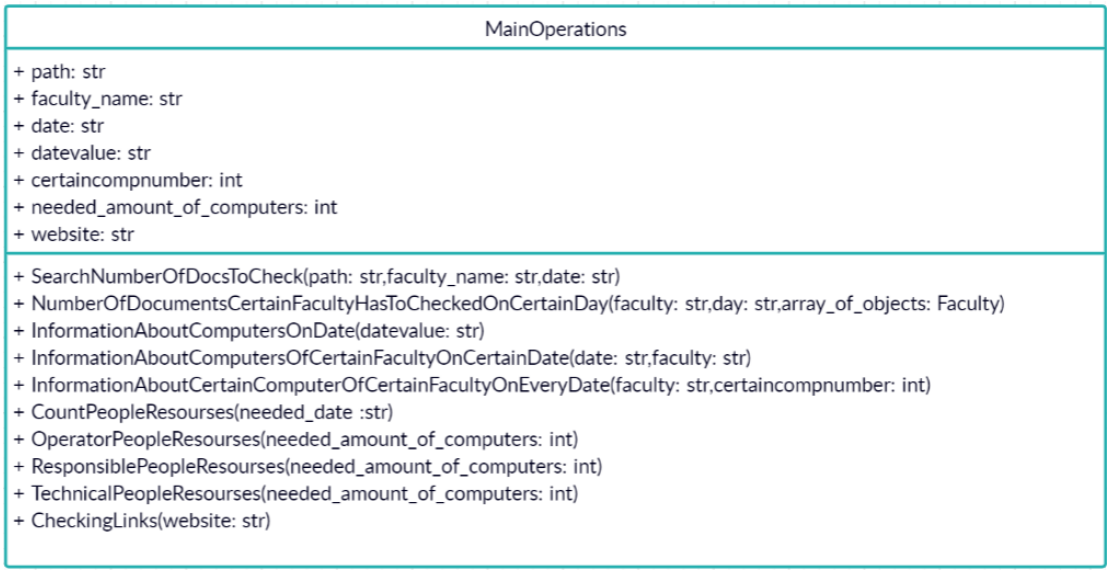

Рисунок 5.7 - Діаграма класу MainOperations

Клас призначений для забезпечення основного функціоналу програми

Таблиця 5.10 – Опис методів класу MainOperations:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def SearchNumberOfDocsToCheck(path, faculty_name,date)	path, faculty_name, date	Value	Функція для повернення значення числа заяв, які необхідно перевірити вказаному факультету у вказану дату
2	def InformationAboutComputersOnDate(datevalue)	Datevalue	final_answer_array	Функція для повернення масиву об’єктів зі

				значеннями параметрів кожного ПК за вказану дату
3	def InformationAboutComputersOfCertainFacultyOnCertainDate(date,faculty)	date,faculty	final_answer_array	Функція для повернення масиву об’єктів зі значеннями параметрів кожного ПК вказаного факультету за вказану дату
4	def InformationAboutCertainComputerOfCertainFacultyOnEveryDate(faculty,certaincompnumber)	faculty,certaincompnumber	final_answer_array	Функція для повернення масиву об’єктів зі значеннями параметрів вказаного ПК вказаного факультету за весь період часу
5	def CountPeopleResourses(needed_date)	needed_date	total_people_resourses_array	Функція для повернення масиву зі значеннями назв факультетів і числа ПК що необхідно задіяти у вказану дату
6	def OperatorPeopleResourses(needed_amount_of_computers)	needed_amount_of_computers	Operators_needed	Функція для повернення масиву значення числа операторів що необхідно задіяти у вказану дату
7	def ResponsiblePeopleResourses(needed_amount_of_computers)	needed_amount_of_computers	ResponsiblePeople_needed	Функція для повернення масиву значення числа відповідальних що необхідно задіяти у вказану дату
8	def TechnicalPeopleResourses(needed_amount_of_computers)	needed_amount_of_computers	TechnicalPeople_needed	Функція для повернення масиву значення числа технічних секретарів що необхідно задіяти у вказану дату
9	Def CheckingLinks(website)	Website	final_answer_array	Функція для повернення масиву значень посилань та їх наявності на сайті

5.3.6. Клас main

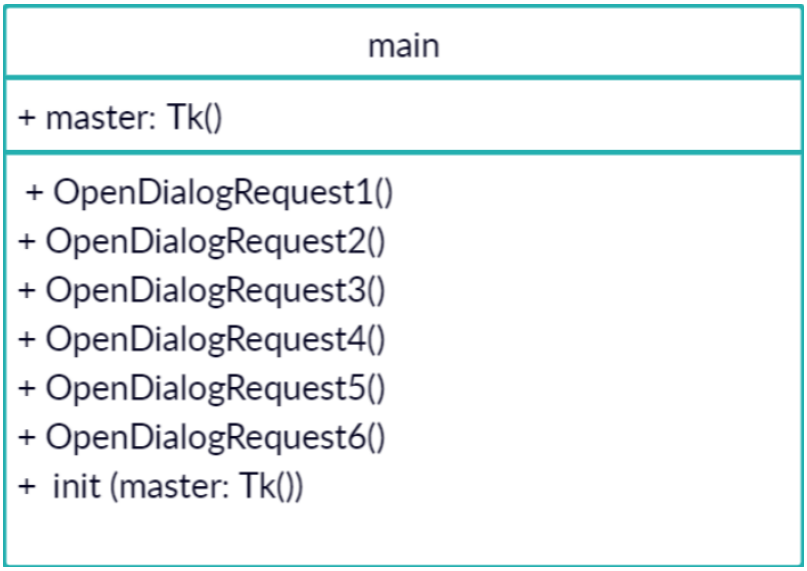

Рисунок 5.8 - Діаграма класу main

Клас створений для опису стартового вікна приграми

Таблиця 5.11 – Опис методів класу main:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, master)	self, master	-	Конструктор за замовчуванням
2	def OpenDialogRequest1(self)	Self – мітка екземпляра	-	Функція для виклику дочірнього вікна
3	def OpenDialogRequest2(self)	Self– мітка екземпляра	-	Функція для виклику дочірнього вікна
4	def OpenDialogRequest3(self)	Self– мітка екземпляра	-	Функція для виклику дочірнього вікна
5	def OpenDialogRequest4(self)	Self– мітка екземпляра	-	Функція для виклику дочірнього вікна
6	def OpenDialogRequest5(self)	Self– мітка екземпляра	-	Функція для виклику дочірнього вікна
7	def OpenDialogRequest6(self)	Self– мітка екземпляра	-	Функція для виклику дочірнього вікна

5.3.7. Клас ChildWindowRequest1

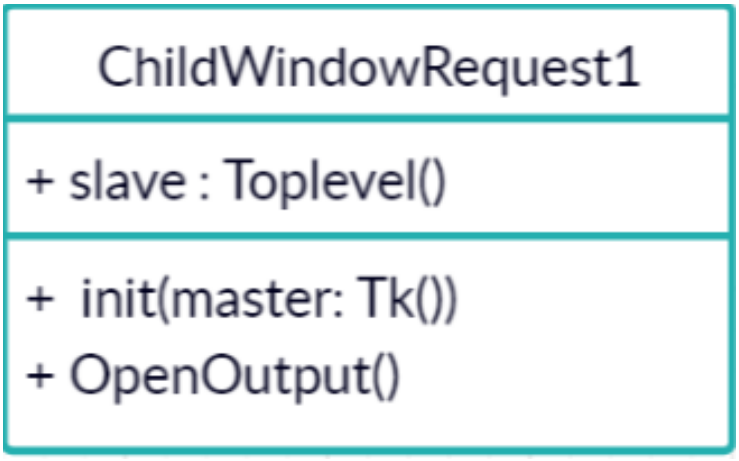

Рисунок 5.9 - Діаграма класу ChildWindowRequest1

Даний клас створений для опису вікна вводу інформації про дату, в разі натискання першої кнопки меню

Таблиця 5.12 – Опис методів класу ChildWindowRequest1:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, master)	self, master	-	Конструктор за замовчуванням
2	def OpenOutput(self)	Self – мітка екземпляра	-	Функція для виклику інформаційного вікна

5.3.8. Клас ChildWindowRequest2

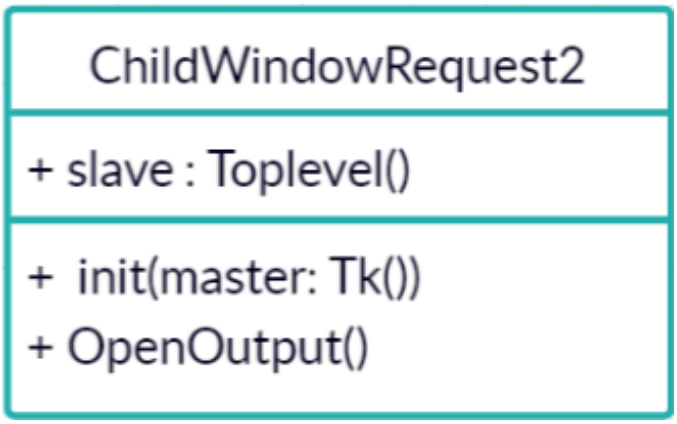

Рисунок 5.10 - Діаграма класу ChildWindowRequest2

Даний клас створений для опису вікна вводу інформації про дату та факультет, в разі натискання другої кнопки меню

Таблиця 5.13 – Опис методів класу ChildWindowRequest2:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, master)	self, master	-	Конструктор за замовчуванням
2	def OpenOutput(self)	Self – мітка екземпляра	-	Функція для виклику інформаційного вікна

5.3.9. Клас ChildWindowRequest3

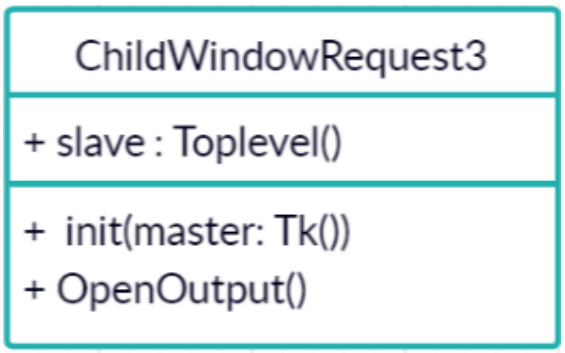

Рисунок 5.11 - Діаграма класу ChildWindowRequest3

Даний клас створений для опису вікна вводу інформації про факультет та порядковий номер комп’ютера, в разі натискання третьої кнопки меню

Таблиця 5.14 – Опис методів класу ChildWindowRequest3:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, master)	self, master	-	Конструктор за замовчуванням
2	def OpenOutput(self)	Self – мітка екземпляра	-	Функція для виклику інформаційного вікна

5.3.10. Клас ChildWindowRequest4

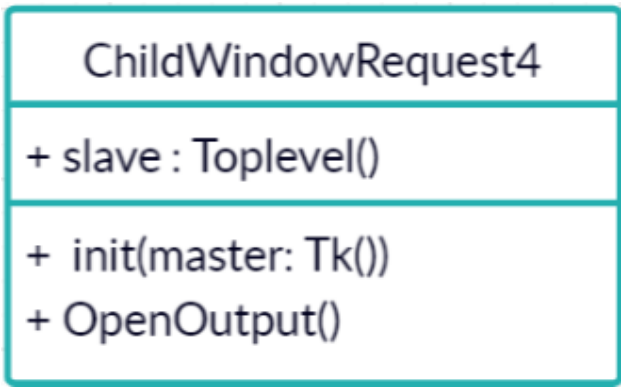

Рисунок 5.12 - Діаграма класу ChildWindowRequest4

Даний клас створений для опису вікна вводу інформації про дату , в разі натискання четвертої кнопки меню

Таблиця 5.15 – Опис методів класу ChildWindowRequest4:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, master)	self, master	-	Конструктор за замовчуванням
2	def OpenOutput(self)	Self – мітка екземпляра	-	Функція для виклику інформаційного вікна

5.3.11. Клас ChildWindowRequest5

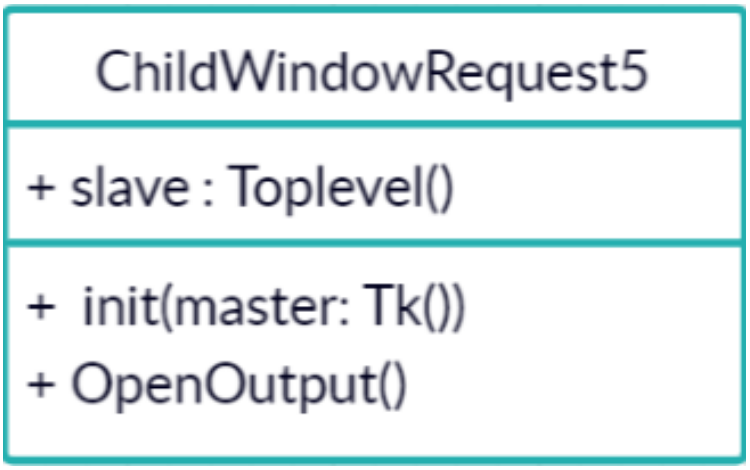

Рисунок 5.13 - Діаграма класу ChildWindowRequest5

Даний клас створений для опису вікна вводу інформації про дату та факультет, в разі натискання п’ятої кнопки меню

Таблиця 5.16 – Опис методів класу ChildWindowRequest5:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, master)	self, master	-	Конструктор за замовчуванням
2	def OpenOutput(self)	Self – мітка екземпляра	-	Функція для виклику інформаційного вікна

5.3.12. Клас ChildWindowRequest6

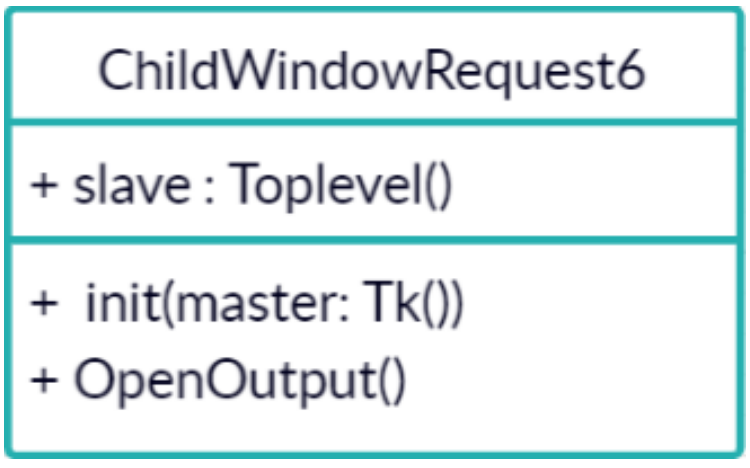

Рисунок 5.14 - Діаграма класу ChildWindowRequest6

Даний клас створений для опису вікна вводу інформації про веб-сайт , в разі натискання шостої кнопки меню

Таблиця 5.17 – Опис методів класу ChildWindowRequest6:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, master)	self, master	-	Конструктор за замовчуванням
2	def OpenOutput(self)	Self – мітка екземпляра	-	Функція для виклику інформаційного вікна

5.3.13. Клас GrandChildWindowRequest1

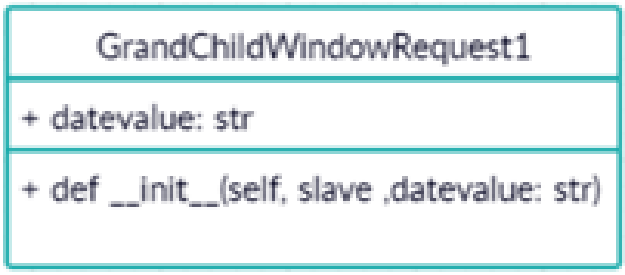

Рисунок 5.15 - Діаграма класу GrandChildWindowRequest1

Даний клас створений для опису вікна виводу інформації про параметри всіх комп'ютерів за вказану дату, в разі натискання першої кнопки меню

Таблиця 5.18 – Опис методів класу GrandChildWindowRequest1:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, slave ,datevalue='')	self, slave, datevalue	-	Конструктор за замовчуванням

5.3.14. Клас GrandChildWindowRequest2

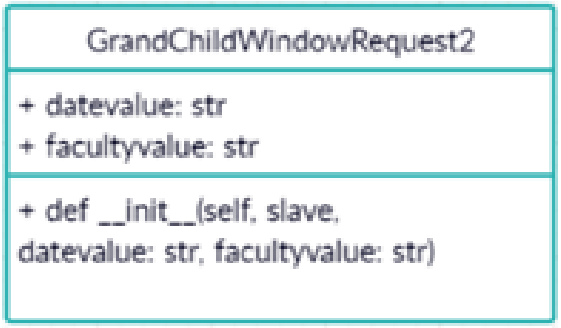

Рисунок 5.16 - Діаграма класу GrandChildWindowRequest2

Даний клас створений для опису вікна вводу інформації про параметри всіх комп'ютерів факультету за вказану дату, в разі натискання другої кнопки меню

Таблиця 5.19 – Опис методів класу GrandChildWindowRequest2:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, slave, datevalue, facultyvalue)	self, slave, datevalue, facultyvalue	-	Конструктор за замовчуванням

5.3.15. Клас GrandChildWindowRequest3

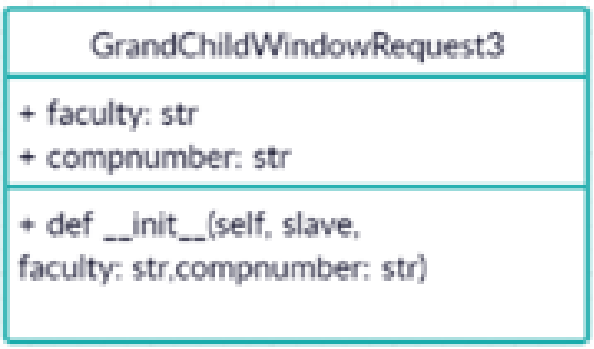

Рисунок 5.17 - Діаграма класу GrandChildWindowRequest3

Даний клас створений для опису вікна вводу інформації про параметри вказаного комп'ютера факультету за весь період часу, в разі натискання третьої кнопки меню

Таблиця 5.20 – Опис методів класу GrandChildWindowRequest3:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, slave, faculty, compnumber)	self, slave, faculty, compnumber	-	Конструктор за замовчуванням

5.3.16. Клас GrandChildWindowRequest4

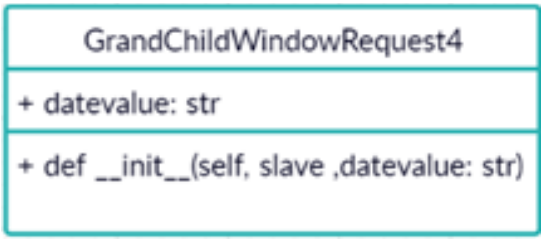

Рисунок 5.18 - Діаграма класу GrandChildWindowRequest4

Даний клас створений для опису вікна виводу інформації про необхідні людські ресурси за вказану дату, в разі натискання четвертої кнопки меню

Таблиця 5.21 – Опис методів класу GrandChildWindowRequest4:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, slave ,datevalue)	self, slave, datevalue	-	Конструктор за замовчуванням

5.3.17. Клас GrandChildWindowRequest5

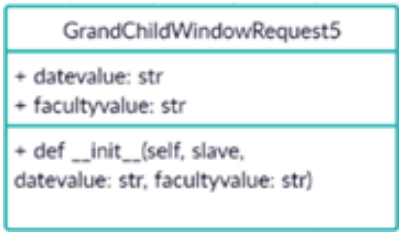

Рисунок 5.19 - Діаграма класу GrandChildWindowRequest5

Даний клас створений для опису вікна виводу інформації про необхідні людські ресурси вказаного факультету за вказану дату, в разі натискання п’ятої кнопки меню

Таблиця 5.22 – Опис методів класу GrandChildWindowRequest5:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, slave, datevalue, facultyvalue)	self, slave, datevalue, facultyvalue	-	Конструктор за замовчуванням

5.3.18. Клас GrandChildWindowRequest6

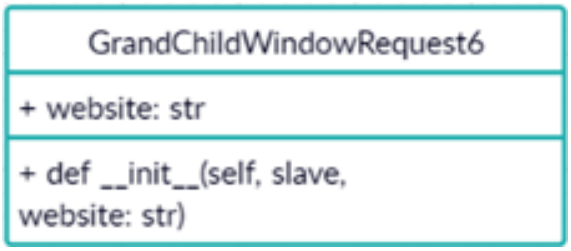

Рисунок 5.20 - Діаграма класу GrandChildWindowRequest6

Даний клас створений для опису вікна виводу інформації про наявність посилань на сайті, в разі натискання четвертої кнопки меню

Таблиця 5.23 – Опис методів класу GrandChildWindowRequest6:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def __init__(self, slave, website)	self, slave, website	-	Конструктор за замовчуванням

5.3.19. Клас ReturnArrayModule

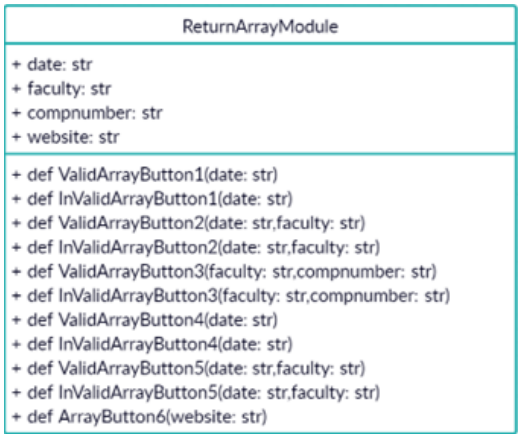

Рисунок 5.21 - Діаграма класу ReturnArrayModule

Даний клас створений для формування масивів табличного виводу інформації

Таблиця 5.24 – Опис методів класу ReturnArrayModule:

№ з/п	Сигнатура	Вхідні параметри	Повернене значення	Призначення
1	def ValidArrayButton1(date)	Date	total_array_with_values_button_number_1	Функція для формування масиву табличного виводу інформації
2	def InValidArrayButton1(date)	Date	total_array_with_invalid_values_button_number_1	Функція для формування масиву табличного виводу інформації
3	def ValidArrayButton2(date, faculty)	date, faculty	total_array_with_values_button_number_2	Функція для формування масиву табличного виводу інформації
4	def InValidArrayButton2(date,faculty)	date, faculty	total_array_with_invalid_values_button_number_2	Функція для формування масиву табличного виводу інформації
5	def ValidArrayButton3(faculty, compnumber)	faculty, compnumber	total_array_with_values_button_number_3	Функція для формування масиву табличного виводу інформації
6	def InValidArrayButton3(faculty, compnumber)	faculty, compnumber	total_array_with_invalid_values_button_number_3	Функція для формування масиву табличного виводу інформації

7	def ValidArrayButton4(date)	Date	total_array_with_values_button_number_4	Функція для формування масиву табличного виводу інформації
8	def InValidArrayButton4(date)	Date	total_array_with_invalid_values_button_number_4	Функція для формування масиву табличного виводу інформації
9	def ValidArrayButton5(date,faculty)	date, faculty	total_array_with_values_button_number_5	Функція для формування масиву табличного виводу інформації
10	def InValidArrayButton5(date,faculty)	date, faculty	total_array_with_invalid_values_button_number_5	Функція для формування масиву табличного виводу інформації
11	def ArrayButton6(website)	website	total_array_with_values_button_number_6	Функція для формування масиву табличного виводу інформації

5.4. Створення і опис роботи програми – агента

Програма-агент призначена для збору необхідних параметрів комп’ютера та встановлюється на кожний ПК відбіркових комісій всіх факультетів. Програма починає свою роботу одночасно з запуском операційної системи і створює файл (Рисунок 5.22 – приклад файлу, який створює агент),в котрий записує значення параметрів. Створений файл копіюється в поточну директорію(на випадок, якщо будуть проблеми зі з’єднанням комп’ютерів ,файл, все одно збережеться) і у спільну мережеву папку основного комп’ютера приймальної комісії (на котрому встановлений основний додаток).

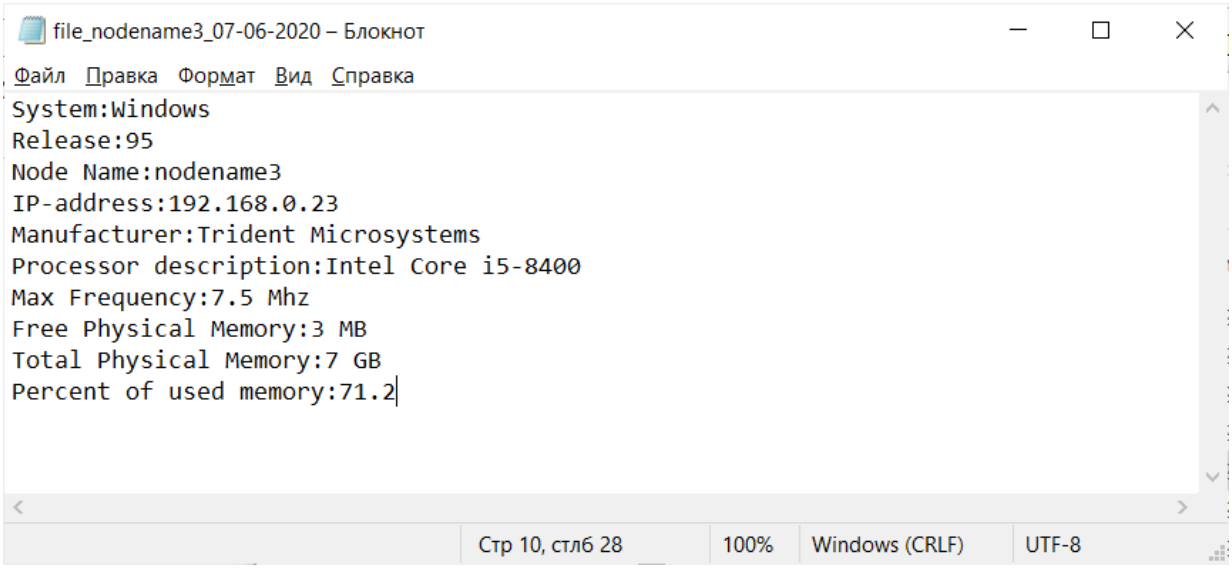

Рисунок 5.22 – приклад файлу, який створює агент

5.5. Висновки до розділу

В даному розділі були розглянуті необхідні програмні модулі, встановлено структуру ПЗ, розроблено класи для реалізації роботи додатку.

6. РОБОТА КОРИСТУВАЧА З ПРОГРАМНОЮ СИСТЕМОЮ

6.1. Інсталяція та системні вимоги

Для роботи створеної програми необхідно завантажити всі розроблені модулі в мережеву папку комп’ютерів, а також програму- агента, на кожний ПК , налагодити з’єднання та обмін файлами з посередництвом комутатора

6.2 Сценарій роботи з програмою

Сценарій роботи з розробленим ПЗ представлений наступною use case діаграмою

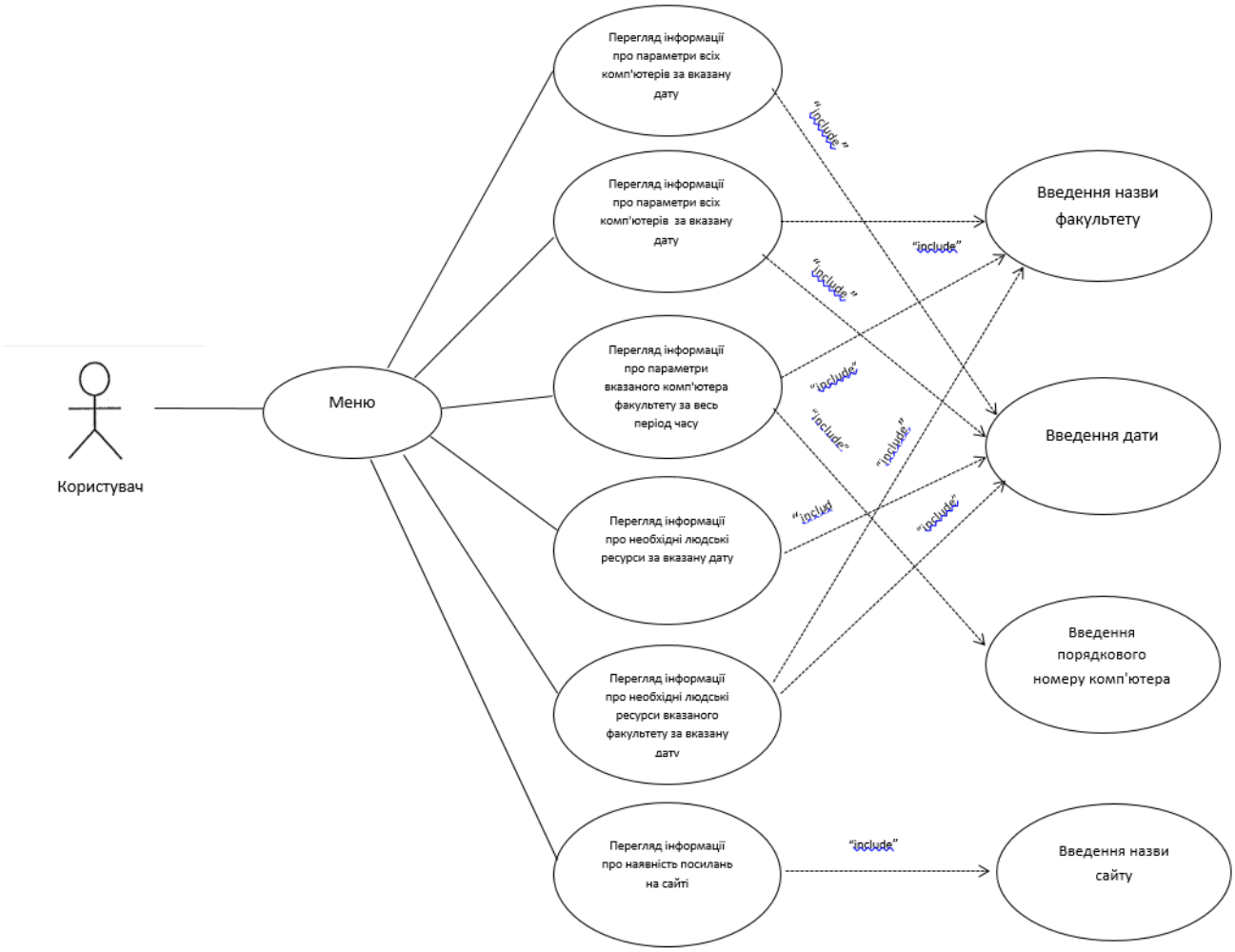

Рисунок 6.1- use case діаграма

6.3. Демонстрація роботи додатку

6.3.1 Демонстраційний приклад номер 1

Одразу після запуску програми користувач бачить на екрані меню програми з вказаними функціями, які виконує додаток

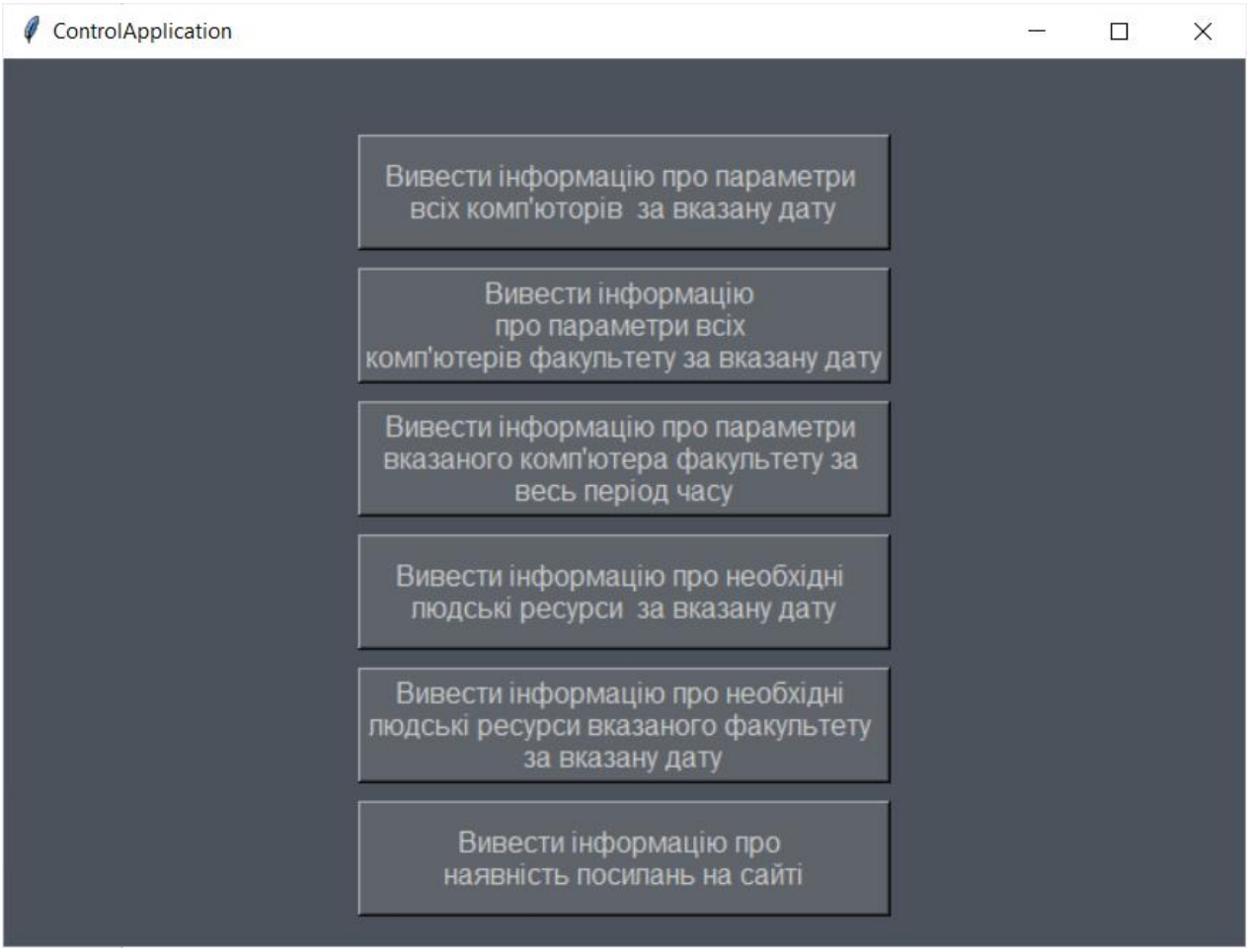

Рисунок 6.2 – Головне вікно ПД з меню

При натисканні на кнопку "Вивести інформацію про параметри всіх комп’ютерів за вказану дату" на екрані з’являється вікно для вводу дати. Користувач може обрати дату з запропонованих

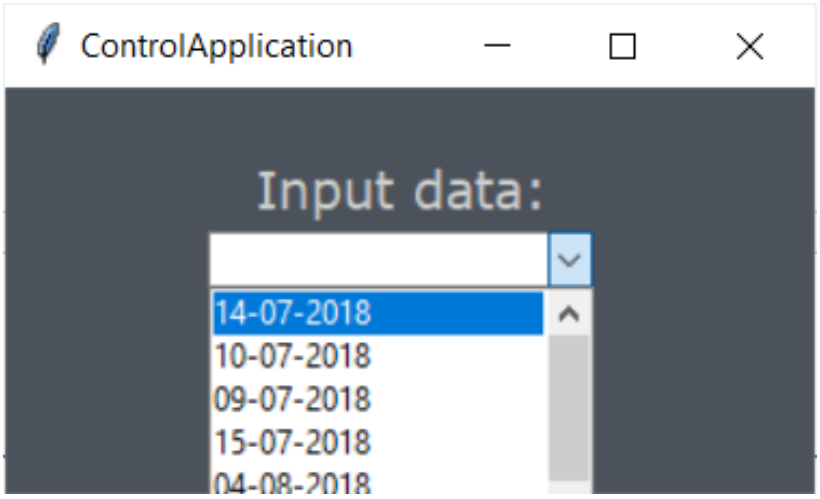

Рисунок 6.2 – Вікно для вводу дати

Далі на екрані з’являється вікно з інформацією про параметри всіх комп’ютерів, та інформацією про помилки і попередження, в разі якщо файл не існує, або виникли проблеми з підключенням

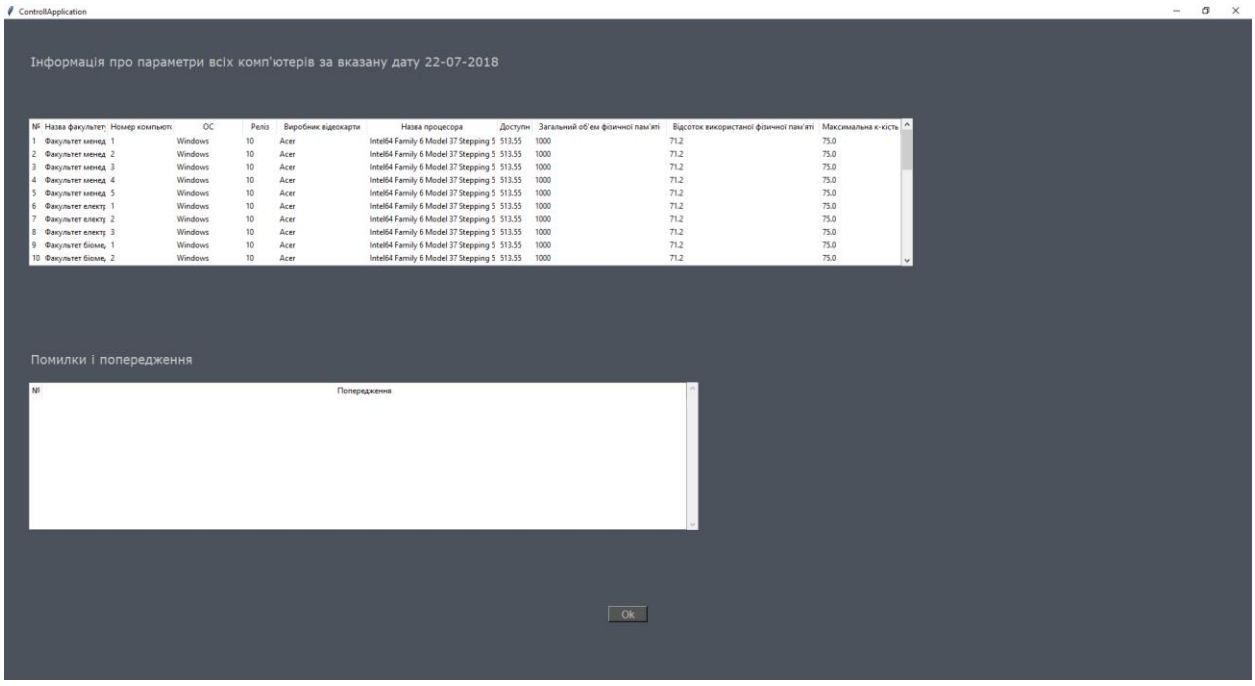

Рисунок 6.4 – Інформаційне вікно з інформацією про параметри всіх комп’ютерів, та інформацією про помилки і попередження

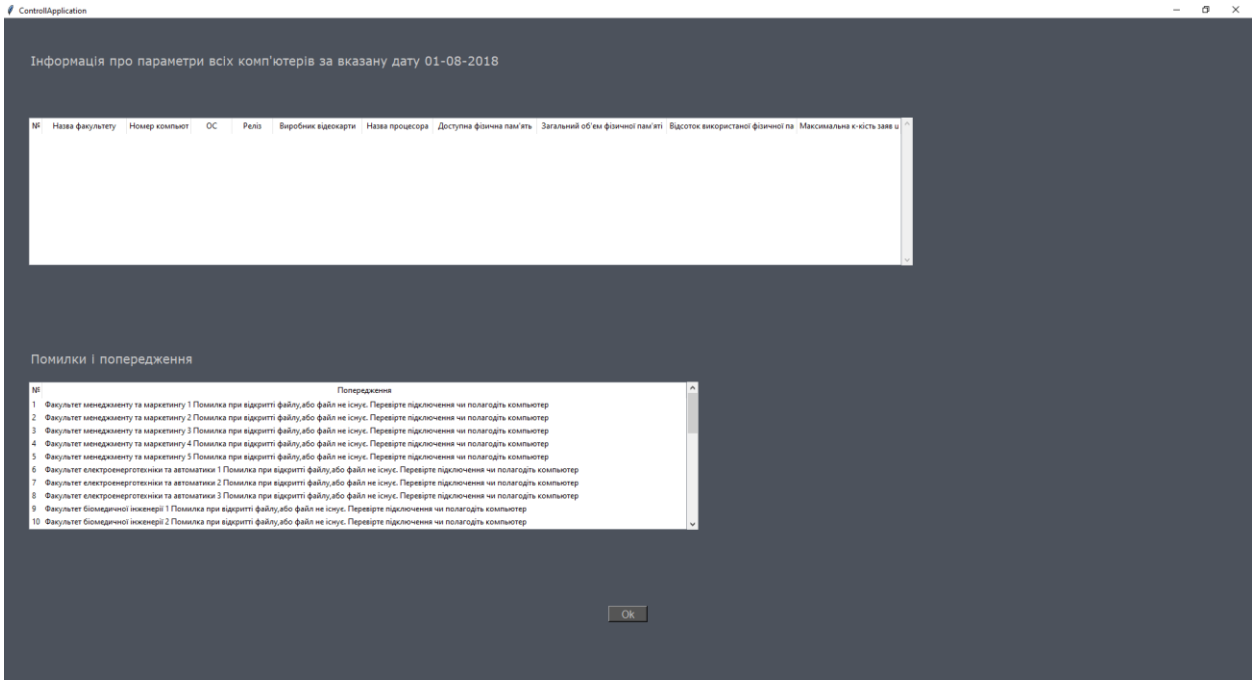

Рисунок 6.5– Інформаційне вікно з інформацією про помилки і попередження

№	Попередження
30	Приладобудівний факультет 1 Помилка при відкритті файлу,або файл не існує. Перевірте підключення чи полагодіть компьютер
31	Приладобудівний факультет 2 Помилка при відкритті файлу,або файл не існує. Перевірте підключення чи полагодіть компьютер
32	Хіміко-технологічний факультет 1 Помилка при відкритті файлу,або файл не існує. Перевірте підключення чи полагодіть компьютер
33	Хіміко-технологічний факультет 2 Помилка при відкритті файлу,або файл не існує. Перевірте підключення чи полагодіть компьютер
34	Факультет біотехнології і біотехніки 1 Помилка при відкритті файлу,або файл не існує. Перевірте підключення чи полагодіть компьютер
35	Факультет біотехнології і біотехніки 2 Помилка при відкритті файлу,або файл не існує. Перевірте підключення чи полагодіть компьютер
36	Видавничо-поліграфічний інститут 1 Помилка при відкритті файлу,або файл не існує. Перевірте підключення чи полагодіть компьютер
37	Видавничо-поліграфічний інститут 2 Помилка при відкритті файлу,або файл не існує. Перевірте підключення чи полагодіть компьютер
38	Інститут енергозбереження та енергоменеджменту 1 Помилка при відкритті файлу,або файл не існує. Перевірте підключення чи полагодіть компьютер
39	Інженерно-хімічний факультет 1 Помилка при відкритті файлу,або файл не існує. Перевірте підключення чи полагодіть компьютер

Рисунок 6.6– Приклад виводу помилок

В кожному рядку помилки зазначено назву факультету, якому належить ПК та його порядковий номер

6.3.2 Демонстраційний приклад номер 2

Після вибору 4-ї кнопки меню та введення дати, на екрані з’явиться інформаційне вікно з виводом інформації про необхідні людські ресурси

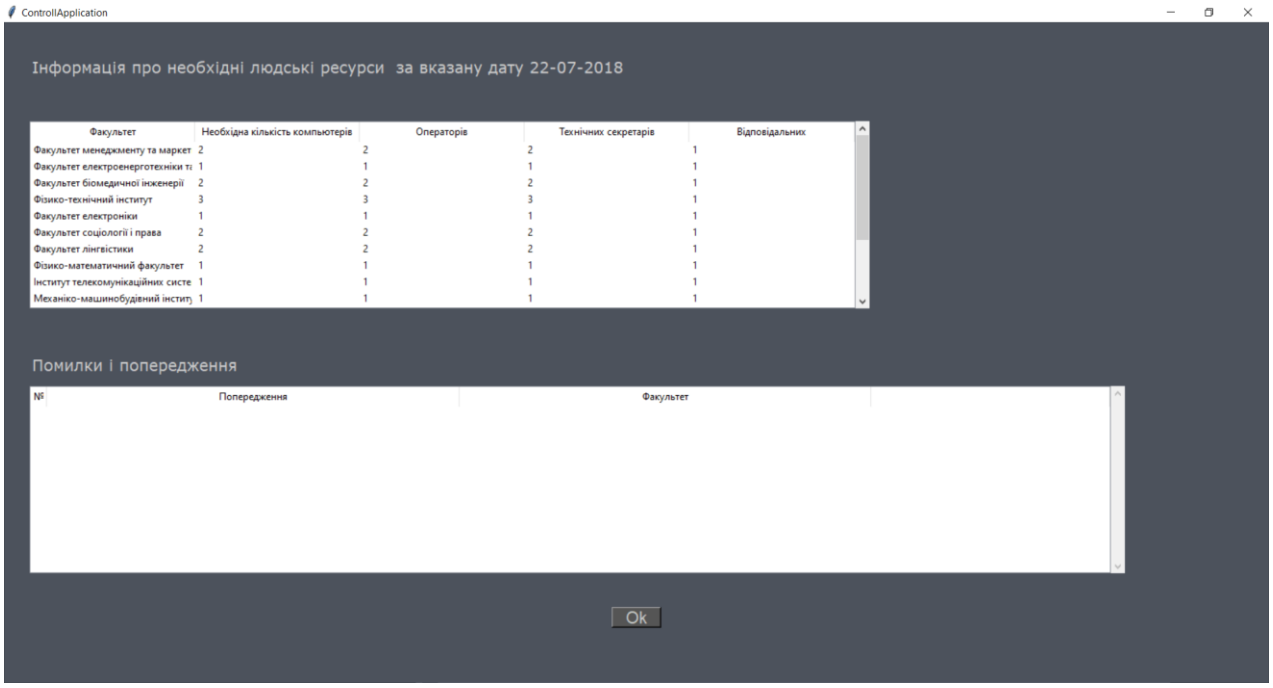

Рисунок 6.7 – Інформаційне вікно з інформацією про необхідні людські ресурси, та інформацією про помилки і попередження

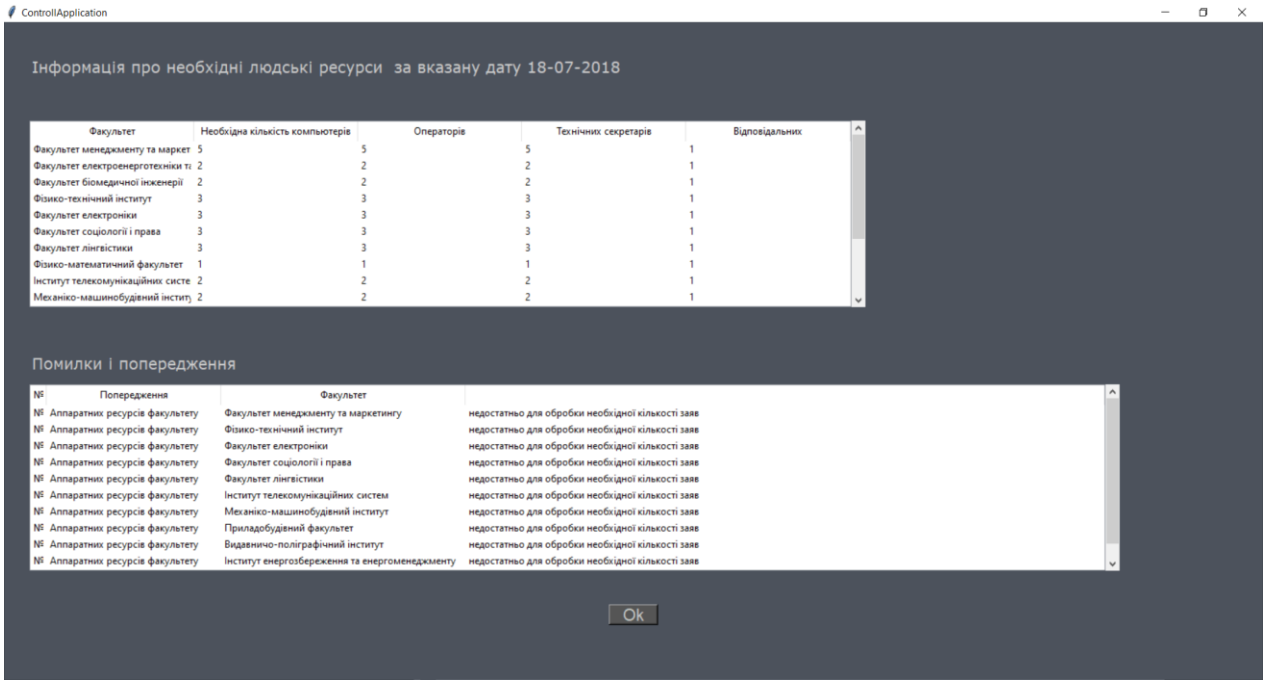

Рисунок 6.8 – Інформаційне вікно з інформацією про помилки і попередження

6.4. Висновки до розділу

В даному розділі було надано інструкції з інсталяції та використання розробленого ПЗ, а також наведені демонстраційні приклади роботи

ВИСНОВКИ

В роботі було розроблено програмне забезпечення для контролю дієздатності відбіркових комісій факультетів та інститутів. Проведено аналіз інформаційного потоку Приймальної комісії протягом останніх років, проаналізувати комп'ютерне забезпечення відбіркових комісій факультетів. В залежності від функціональних можливостей комп'ютерного устаткування, кількості поданих заяв абітурієнтів визначено план виходу на роботу технічних співробітників відбіркових комісій факультетів та кількості комп'ютерів, необхідних для своєчасної обробки інформації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Положення про Приймальну комісію Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» – [Електронний ресурс] – Режим доступу до ресурсу: http://pk.kpi. ua/wp-content/uploads/2016/12/admission-comettee-2017.pdf

2. Правила прийому до Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського в 2018 році» – [Електронний ресурс] – Режим доступу до ресурсу: http://pk.kpi.ua/wp-content/uploads/2019/02/rules-2019.pdf

3. IT блог: программы и руководства – [Електронний ресурс] – Режим доступу до ресурсу: https://lyapidov.ru/two-pc-in-lan-with-twisted-pair/

4. Лутц М.Изучаем Python, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с.

5. Функциональность PyCharm– [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com/ru-ru/pycharm/features/

6. PyCharm Community Edition and Professional Edition Explained: Licenses and More- [Електронний ресурс] – Режим доступу до ресурсу: https://blog.jetbrains.com/pycharm/2017/09/pycharm-community-edition-and-professional-edition-explained-licenses-and-more/

7. PyCharm: the Python IDE for Professional Developers by JetBrains- [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com/pycharm/

8. Модуль datetime– [Електронний ресурс] – Режим доступу до ресурсу: https://all-python.ru/osnovy/modul-datetime.html

9. Пособие по MySQL на Python– [Електронний ресурс] – Режим доступу до ресурсу: https://www.internet-technologies.ru/articles/posobie-po-mysql-na-python.html

10. Конспект по Python. Модуль Tkinter (часть 2) – [Електронний ресурс] – Режим доступу до ресурсу: http://www.239.ru/sites/default/files/userdata/konspekt12._modul_tkinter_chast2.pdf

11. Работа с файлами в Python с помощью модуля OS– [Електронний ресурс] – Режим доступу до ресурсу: https://pythonru.com/osnovy/rabota-s-fajlami-v-python-s-pomoshhju-modulja-os

12. Билл Любанович/ Простой Python. Современный стиль программирования// "Издательский дом ""Питер""", 7 июн. 2016 г.

13. Мэтиз Эрик/ Изучаем Python. Программирование игр, визуализация данных, веб-приложения. 2-е изд.// "Издательский дом ""Питер""", 28 апр. 2017 г

14. https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:Star_topology.PNG

ДОДАТОК А

Автоматизована система контролю працездатності ресурсів Приймальної комісії КПІ ім. Ігоря Сікорського

Специфікація

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61126_20Б

Аркушів 1

Київ 2020

Позначення	Найменування	Примітки
	Документація	
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61126_20Б	Записка.docx	Текстова частина дипломної роботи
	Компоненти	
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61126_20Б	AppFile.py	Файл з визначенням класу ReturnArrayModule
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61126_20Б	Faculty.py	Файл з визначенням класів Faculty, Faculty_computer, ConstantValues
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61126_20Б	WindowsPackaging.py	Файл запуску додатку
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61126_20Б	Operations.py	Файл з визначенням операцій
УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61126_20Б	Agent.py	Файл програми – агента

ДОДАТОК Б

Автоматизована система контролю працездатності ресурсів Приймальної комісії КПІ ім. Ігоря Сікорського

Лістинг програми

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61126_20Б

Аркушів 10

Київ 2020

```
# Лістинг модулю Agent.py
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# vim:fileencoding=utf-8
import platform
import wmi
import psutil
import datetime
import socket
import schedule
import time

def get_size(bytes, suffix="B"):
    factor = 1024
    for unit in ["", "K", "M", "G", "T", "P"]:
        if bytes < factor:
            return f"{bytes:.2f}{' '}{unit}{suffix}"
        bytes /= factor

def execute_parameters():
    #назва та версія операційної системи
    uname = platform.uname()
    system_name=uname.system
    release=uname.release
    node_name=uname.node
    #ip адреса
    ip_address=socket.gethostbyname(socket.gethostname())
```

```
#відеокарта в залежності від фірми-виробника
computer = wmi.WMI()
computer_info = computer.Win32_ComputerSystem()[0]
manufacturer_info=computer_info.Manufacturer
#назва процессора
proc_info = computer.Win32_Processor()[0]
proc_description=proc_info.Description
#максимальна тактова частота процессора
cpufreq = psutil.cpu_freq()
max_frequency=cpufreq.max
#оперативна пам'ять(доступна фізична пам'ять)
svmem = psutil.virtual_memory()
free_physical_memory=get_size((int((svmem.available))))
#об'єм жорсткого диску(повний об'єм фізичної пам'яті)
total_physical_memory=get_size((int((svmem.total))))
#об'єм використаної фізичної пам'яті у відсотках
percent_of_used_memory=svmem.percent


#реалізація функції створення та запису до файлу
now = datetime.datetime.now()
file = open("file_"+node_name+"_" + now.strftime("%d-%m-%Y") + ".txt", "w")
file.write('System:'+system_name + '\n')
file.write('Release:'+release + ' \n')
file.write('Node Name:'+node_name+ ' \n')
file.write('IP-address:'+ip_address+ ' \n')
file.write('Manufacturer:'+manufacturer_info+ ' \n')
file.write('Processor description:'+proc_description + ' \n')
```

```
    file.write('Max Frequency:'+str(max_frequency)+' Mhz'+' \n')
    file.write('Free Physical Memory:'+free_physical_memory+ ' \n')
    file.write('Total Physical Memory:'+total_physical_memory+ ' \n')
    file.write('Percent of used memory:'+str(percent_of_used_memory))
    file.close()

schedule.every(2).minutes.do(execute_parameters)
while True:
   schedule.run_pending()
   time.sleep(1)



# Лістинг модулю ReturnArrayModule.py
import Operations
import itertools
class ReturnArrayModule:
    def ValidArrayButton4(date):
        array_to_print=Operations.MainOperations.CountPeopleResourses(date)
        array_valid_values_button_number_4=[]
        array_invalid_values_button_number_4=[]
        for i in range(len(array_to_print)):
            if (array_to_print[i][1])>=0:
                array_valid_values_button_number_4.append(array_to_print[i])
            else:
                array_invalid_values_button_number_4.append(array_to_print[i])
        total_array_to_put_into_frame=[]
        array_to_put_into_frame=[]
        for i in range(len(array_to_print)):
```

```
            array_to_put_into_frame=[]
            array_to_put_into_frame.append(array_to_print[i][0])
            if abs(abs(array_to_print[i][1]))!=0:
                array_to_put_into_frame.append(abs(array_to_print[i][1]))
            else:
                array_to_put_into_frame.append('    ')

array_to_put_into_frame.append(Operations.MainOperations.OperatorPeopleResourses(a
bs(array_to_print[i][1])))

array_to_put_into_frame.append(Operations.MainOperations.TechnicalPeopleResourses(a
bs(array_to_print[i][1])))

array_to_put_into_frame.append(Operations.MainOperations.ResponsiblePeopleResourse
s(abs(array_to_print[i][1])))
            total_array_to_put_into_frame.append(array_to_put_into_frame)
        return total_array_to_put_into_frame
    def InValidArrayButton4(date):
        array_to_print=Operations.MainOperations.CountPeopleResourses(date)
        array_valid_values_button_number_4=[]
        array_invalid_values_button_number_4=[]
        for i in range(len(array_to_print)):
            if (array_to_print[i][1])>=0:
                array_valid_values_button_number_4.append(array_to_print[i])
            else:
                array_invalid_values_button_number_4.append(array_to_print[i])
        total_array_to_put_into_frame=[]
        array_to_put_into_frame=[]
        for i in range(len(array_to_print)):
            if (array_to_print[i][1])==0:
```

```
            array_to_put_into_frame=[]
            array_to_put_into_frame.append(i+1)
            array_to_put_into_frame.append("Не знайдено файлів з")
            array_to_put_into_frame.append("інформацією про параметри ПК")
            array_to_put_into_frame.append("неможливо обчислити кількість
працівників")
            total_array_to_put_into_frame.append(array_to_put_into_frame)
            break


        if (array_to_print[i][1])<0:
            array_to_put_into_frame=[]
            array_to_put_into_frame.append(i+1)
            array_to_put_into_frame.append("Аппаратних ресурсів факультету ")
            array_to_put_into_frame.append(array_to_print[i][0])
            array_to_put_into_frame.append("недостатньо для обробки необхідної
кількості заяв")
            total_array_to_put_into_frame.append(array_to_put_into_frame)
    return total_array_to_put_into_frame
def InValidArrayButton5(date,faculty):
    final_array_to_print=[]
    array_with_all_faculties=ReturnArrayModule.InValidArrayButton4(date)
    final_array_to_print.append(array_with_all_faculties)
    return array_with_all_faculties
def ValidArrayButton5(date,faculty):
    final_array_to_print=[]
    array_to_put_into_frame=[]
    array_to_put_into_frame.append('Факультет')
    array_to_put_into_frame.append('Необхідна кількість компьютерів')
    array_to_put_into_frame.append('Операторів')
```

```
        array_to_put_into_frame.append('Технічних секретарів')
        array_to_put_into_frame.append('Відповідальних')
        array_with_all_faculties=ReturnArrayModule.ValidArrayButton4(date)
        for i in range(len(array_with_all_faculties)):
            for j in range(len(array_with_all_faculties[i])):
                if array_with_all_faculties[i][j]== faculty:
                    array_to_put_into_frame=[]
                    final_array_to_print.append(array_with_all_faculties[i])
        return final_array_to_print
    def InValidArrayButton1(date):
        array_button_number_1=[]
        total_array_with_invalid_values_button_number_1=[]
        array_invalid_values_button_number_1=[]
        array_invalid_values_button_number_1.append('№')
        array_invalid_values_button_number_1.append('Попередження')

array_button_number_1=Operations.MainOperations.InformationAboutComputersOnDate
(date)
        for i in range(len(array_button_number_1[1])):
            array_invalid_values_button_number_1=[]
            array_invalid_values_button_number_1.append(i+1)

array_invalid_values_button_number_1.append(str(array_button_number_1[1][i].get_facu
lty_name())+' '+

str(array_button_number_1[1][i].get_number_of_compuer())+' '+
                                                "Помилка при відкритті файлу,або файл не існує.
Перевірте підключення чи полагодіть компьютер")

total_array_with_invalid_values_button_number_1.append(array_invalid_values_button_
number_1)
```

```
        return total_array_with_invalid_values_button_number_1
    def ValidArrayButton1(date):
        array_button_number_1=[]
        total_array_with_values_button_number_1=[]
        array_valid_values_button_number_1=[]
        array_valid_values_button_number_1.append('№')
        array_valid_values_button_number_1.append('Назва факультету')
        array_valid_values_button_number_1.append('Номер компьютора')
        array_valid_values_button_number_1.append('ОС')
        array_valid_values_button_number_1.append('Реліз')
        array_valid_values_button_number_1.append('Виробник відеокарти')
        array_valid_values_button_number_1.append('Назва процесора')
        array_valid_values_button_number_1.append("Доступна фізична пам'ять")
        array_valid_values_button_number_1.append("Загальний об'ем фізичної пам'яті")
        array_valid_values_button_number_1.append("Відсоток використаної фізичної
пам'яті")
        array_valid_values_button_number_1.append("Максимальна к-кість заяв що може
обробити компьютер")

array_button_number_1=Operations.MainOperations.InformationAboutComputersOnDate
(date)
        for i in range(len(array_button_number_1[0])):
            if array_button_number_1[0][i] != 'error':
                array_valid_values_button_number_1=[]
                array_valid_values_button_number_1.append(i+1)

array_valid_values_button_number_1.append(array_button_number_1[0][i].get_faculty_n
ame())

array_valid_values_button_number_1.append(array_button_number_1[0][i].get_number_
of_compuer())
```

```
            (array_button_number_1[0][i].set_os_name('Windows'))

array_valid_values_button_number_1.append(array_button_number_1[0][i].get_os_name(
))

array_valid_values_button_number_1.append(array_button_number_1[0][i].get_os_releas
e())

array_valid_values_button_number_1.append(array_button_number_1[0][i].get_manufact
urer())

array_valid_values_button_number_1.append(array_button_number_1[0][i].get_proceccor
_name())

array_valid_values_button_number_1.append(array_button_number_1[0][i].get_free_phys
ical_memory())

array_valid_values_button_number_1.append(array_button_number_1[0][i].get_total_phy
sical_memory())

array_valid_values_button_number_1.append(array_button_number_1[0][i].get_percent_o
f_used_memory())

array_valid_values_button_number_1.append(array_button_number_1[0][i].get_available
_amount_of_docs_to_check())

total_array_with_values_button_number_1.append(array_valid_values_button_number_1)
        return total_array_with_values_button_number_1


    def ValidArrayButton2(date,faculty):
        array_button_number_2=[]
        array_values_button_number_2=[]
        total_array_with_values_button_number_2=[]
        array_button_number_2=[]
```

```
        total_array_with_values_button_number_2=[]
        array_valid_values_button_number_2=[]

array_button_number_2=Operations.MainOperations.InformationAboutComputersOfCertainFacultyOnCertainDate(date,faculty)
        for i in range(len(array_button_number_2[0])):
            if array_button_number_2[0][i] != 'error':
                array_valid_values_button_number_2.append(i+1)

array_valid_values_button_number_2.append(array_button_number_2[0][i].get_faculty_name())

array_valid_values_button_number_2.append(array_button_number_2[0][i].get_number_of_compuer())

array_valid_values_button_number_2.append(array_button_number_2[0][i].get_os_name())

array_valid_values_button_number_2.append(array_button_number_2[0][i].get_os_release())

array_valid_values_button_number_2.append(array_button_number_2[0][i].get_manufacturer())

array_valid_values_button_number_2.append(array_button_number_2[0][i].get_proceccor_name())

array_valid_values_button_number_2.append(array_button_number_2[0][i].get_free_physical_memory())

array_valid_values_button_number_2.append(array_button_number_2[0][i].get_total_physical_memory())

array_valid_values_button_number_2.append(array_button_number_2[0][i].get_percent_of_used_memory())
```

```
array_valid_values_button_number_2.append(array_button_number_2[0][i].get_available
_amount_of_docs_to_check())

total_array_with_values_button_number_2.append(array_valid_values_button_number_2)
            array_valid_values_button_number_2=[]
        return total_array_with_values_button_number_2



    def InValidArrayButton2(date,faculty):
        array_button_number_2=[]
        total_array_with_invalid_values_button_number_2=[]
        array_invalid_values_button_number_2=[]
        array_invalid_values_button_number_2.append('№')
        array_invalid_values_button_number_2.append('Попередження')

array_button_number_2=Operations.MainOperations.InformationAboutComputersOfCert
ainFacultyOnCertainDate(date,faculty)
        for i in range(len(array_button_number_2[1])):
            array_invalid_values_button_number_2=[]
            array_invalid_values_button_number_2.append(i+1)


…
```

ДОДАТОК В

Автоматизована система контролю працездатності ресурсів Приймальної комісії КПІ ім. Ігоря Сікорського

Опис програмного коду

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61126_20Б

Аркушів 8

Київ 2020

АНОТАЦІЯ

Дана програмна система написана мовою програмування Python за допомогою інтегрованого середовища PyCharm. За допомогою розробленого ПД розвязується задача автоматиззації контролю працездатності ресурсів Приймальної комісії КПІ ім. Ігоря Сікорського. Система може використовуватись в установах,подібних за структурою і апаратним забезпеченням до Приймальної комісії КПІ ім. Ігоря Сікорського.

ЗМІСТ

ЗАГАЛЬНІ ВІДОМОСТІ І ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Дана програмна система була розроблена з метою автоматизації контролю працездатності ресурсів Приймальної комісії КПІ ім. Ігоря Сікорського. Програма має назву ControlApplication і була написана мовою програмування Python за допомогою інтегрованого середовища PyCharm.

За допомогою програмної системи розв'язується наступні задачі:

● Розрахунок кількості заяв, поданих абітурієнтами на конкретний факультет/інститут за конкретну дату виходячи з даних вхідного файлу

● Збір інформації про параметри (ресурси) кожного ПК

● Розрахунок кількості поданих заяв, які спроможні обробити ПК відбіркових комісій конкретного факультету/інституту за конкретну дату, виходячи з вхідного файлу з інформацією про ресурси даного ПК

● Розрахунок та виведення інформації про спроможність обробити вхідну кількість заяв кожним факультетом/інститутом, виходячи з обробки даних про параметри ПК, з загальною кількістю заяв, які подані на даний факультет/інститут, а також інформація про помилки та попередження в разі їх наявності

ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Логічна структура програмного додатку представлена наступною UML-діаграмою

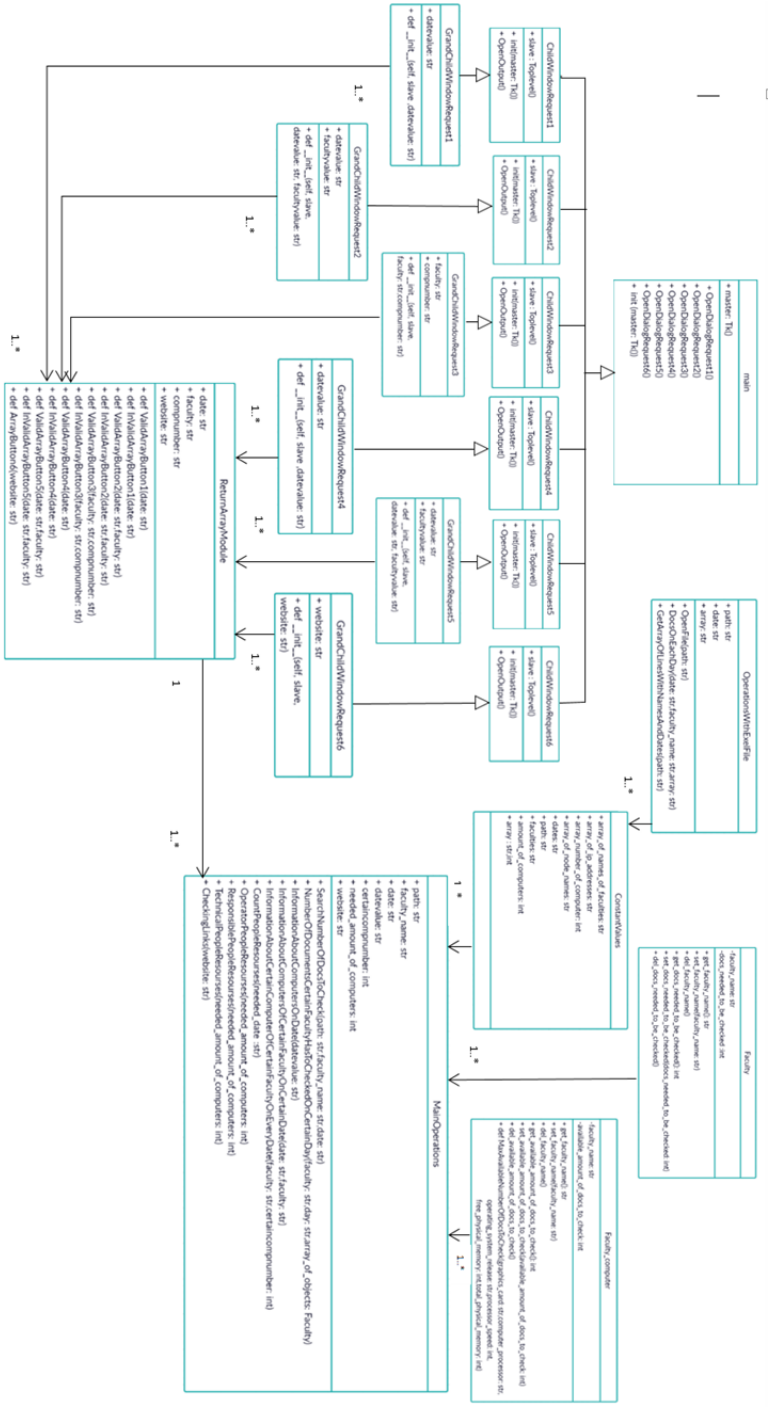

Рисунок В.2.1 - Логічна структура ПД UML-діаграма

ВИКОРИСТОВУВАНІ ТЕХНІЧНІ ЗАСОБИ

Дана програмна система була розроблена саме для використання в Приймальній комісії КПІ ім. Ігоря Сікорського, тому для свого функціонування, окрім самих ПК , вимагає також наявність пристроїв фізичної топології «Зірка» ,а саме – концентратора і кабелю (крученої пари)

ВИКЛИК І ЗАВАНТАЖЕННЯ

Для роботи створеної програми необхідно завантажити всі розроблені модулі в мережеву папку комп’ютерів, а також програму- агента, на кожний ПК , налагодити з’єднання та обмін файлами з посередництвом комутатора Для виклику програми необхідно подвійним кліком миші обрати модуль WindowsPackaging.py, далі перед користувачем на екрані з’явиться меню з вказаними функціями, які виконує додаток

ВХІДНІ І ВИХІДНІ ДАНІ

Вхідним файлом до програми є Excel –файл , в якому містяться дані про дату, час та назву факультету/інституту кожної заяви, яка була подана абітурієнтами в КПІ ім. Ігоря Сікорського

Вихідними даними для створеної системи є таблиці з інформацією про параметри всіх комп’ютерів, інформації про необхідні ресурси, а також попередження і помилки в разі їх наявності

ДОДАТОК Г

Автоматизована система контролю працездатності ресурсів Приймальної комісії КПІ ім. Ігоря Сікорського

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61126_20Б

Актів про впровадження

Аркушів 1

Київ 2020

ЗАТВЕРДЖУЮ
Заступник голови Приймальної комісії
КПІ імені Ігоря Сікорського
Валерій Можаровський
12.05 2020 р.

АКТ

Впровадження матеріалів бакалаврської роботи студентки
Теплоенергетичного факультету Горбатенко Ксенії Вікторівни

Даним актом посвідчується те, що результати бакалаврської роботи Горбатенко К.В. впроваджені в робочий процес Приймальної комісії Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» та використовуються при автоматизації робочих місць Відбіркових комісій факультетів.

Заступник відповідального секретаря
Приймальної комісії Ольга Залевська

ДОДАТОК Д

Автоматизована система контролю працездатності ресурсів Приймальної комісії КПІ ім. Ігоря Сікорського

Публікції

УКР.НТУУ”КПІ”_ТЕФ_АПЕПС_ТР61_№61126_20Б

Аркушів 2

Київ 2020

Міністерство освіти і науки України
Українська асоціація з прикладної геометрії
Мелітопольський державний педагогічний університет
імені Богдана Хмельницького
Мелітопольська школа прикладної геометрії

СУЧАСНІ ПРОБЛЕМИ МОДЕЛЮВАННЯ

ЗБІРНИК НАУКОВИХ ПРАЦЬ

Випуск 16

м. Мелітополь

УДК 657.1+336+514

НЕДОЛІКИ ТА ПЕРЕВАГИ АВТОМАТИЗОВАНИХ СИСТЕМ КОНТРОЛЮ РЕСУРСІВ

Ванін В.В., д.т.н.,
Залевська О.В., к.т.н.,
Горбатенко К.В.
Національний технічний університет України Київський політехнічний інститут імені Ігоря Сікорського (Україна),
Спірінцев Д.В., к.т.н.
Мелітопольський державний педагогічний університет імені Богдана Хмельницького (Україна)

В роботі розглядаються автоматизовані системи (АС) контролю використання ресурсів об'єкту та запропоновано схему класифікації таких систем на основі аналізу їх недоліків та переваг. Автоматизовані системи контролю охоплюють досить різні об'єкти за своїм характером, побудовою, структурою та змістом. Саме з цим пов'язана складність об'єднання всіх автоматизованих систем в єдину структуру. На даний час не існує затвердженої універсальної класифікації автоматизованих систем контролю. Існуючі класифікації будуються на властивостях та ознаках самих систем. Така класифікація не надає повне уявлення про їх організацію, інтегрованість з іншими підсистемами, оцінки та аналізу ефективності їх впровадження таких систем. Вони містять інформацію про вузьку сферу впровадження автоматизованих систем контролю ресурсів, наприклад: рівень або сфера діяльності, рівень автоматизації процесів управління, ступінь централізації обробки інформації та за ступінь інтеграції функцій. Для різних підприємств, для кожної сфери обслуговування, як правило, розробляють власну автоматизовану систему. Такі системи мають включати в себе аналіз, збір та обробку інформації про досліджуваний об'єкт та систем, що з ним пов'язані. В процесі впровадження таких систем постає питання ресурсного забезпечення об'єкта спеціальною технікою, що могла б забезпечити функціонування системи без помилок під час зібрання інформації, оскільки це має відбуватись постійно.

Запропонована структура класифікації автоматизованих систем на основі їх недоліків та переваг створює можливості для