МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА СИСТЕМИ ВІДДАЛЕНОГО ДОСТУПУ»**

Виконав: студент 4 курсу, групи 6.1219-1пі
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми програмна інженерія
(назва освітньої програми)
Д.А. Іванов
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, доцент, к.т.н. Мухін В.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач кафедри фундаментальної та прикладної математики, професор, д.т.н. Гребенюк С.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти бакалавр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма програмна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ 07 ” 02 2023 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Іванову Данилу Артемовичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Розробка системи віддаленого доступу

керівник роботи Мухін Віталій Вікторович, к.т.н., доцент
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 26 » січня 2023 року № 102-с

2. Строк подання студентом роботи 07.06.2023 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Розробка системи віддаленого доступу.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
презентація

6. Консультанти розділів роботи

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

7. Дата видачі завдання 07.02.2023 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	08.02.2023	
2.	Збір вихідних даних.	22.02.2023	
3.	Обробка методичних та теоретичних	21.03.2023	
	джерел.		
4.	Розробка першого та другого розділу.	20.04.2023	
5.	Розробка третього розділу.	15.05.2023	
6.	Оформлення та нормоконтроль	01.06.2023	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	21.06.2023	

Студент _____ Д.А. Іванов
(підпис) (ініціали та прізвище)

Керівник роботи _____ В.В. Мухін
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Розробка системи віддаленого доступу»: 119 с., 7 рис., 11 джерел, 2 додатки.

АРХІТЕКТУРА, ВИРОБНИЦТВО, ГНУЧКІСТЬ, КОНСОЛЬ ОПЕРАЦІЙНОЇ СИТСТЕМИ, МАСШТАБОВАНІСТЬ, РОЗРОБКА, СИСТЕМА ВІДДАЛЕНОГО ДОСТУПУ.

Об’єкт дослідження – розробка системи віддаленого доступу.

Мета роботи: розробити та ввести в експлуатацію систему віддаленого доступу.

Методи дослідження: теоретичні: теоретико-методологічний аналіз; емпіричні: спостереження, тестування, аналіз програмного забезпечення, проєктування.

Наукова новизна полягає у розробці та впровадженні системи віддаленого доступу яка буде простою у використанні для користувачів, гнучкою – здатною до використання у різних сферах ІТ виробництва, масштабованою – здатною до збільшення користувачів.

SUMMARY

Bachelor’s Qualification Paper «Development of a Remote Access System»: 119 p., 7 figures, 11 references, 2 supplements.

ARCHITECTURE, PRODUCTION, FLEXIBILITY, OPERATING SYSTEM CONSOLE, SCALABILITY, DEVELOPMENT, REMOTE ACCESS SYSTEM.

The object of the study is development of remote access system.

The aim of the study is develop and put into operation a remote access system.

The methods of research are theoretical: theoretical and methodological analysis; empirical: observation, testing, software analysis, design.

The scientific novelty is the development and implementation of a remote access system that will be easy to use for users, flexible – capable of use in various areas of IT production, scalable – capable of increasing users.

ЗМІСТ

ВСТУП

За останні роки зросла потреба в можливості працювати з будь-якого місця. Системи віддаленого доступу дозволяють користувачам отримувати доступ до свого комп’ютера або робочого середовища з віддаленої локації. Це особливо важливо для людей, які часто подорожують або потребують працювати з дому.

Однак, замість безкоштовних сервісів, IT компанії вимушені оформлювати підписки на дорогі ресурси, що створює додаткові витрати. Це може бути особливо проблематичним для маленьких бізнесів або окремих користувачів, які не можуть собі дозволити витрати на платну підписку.

Тому, щоб вирішити дану проблему, потрібно розробити систему віддаленого доступу, яка була б доступною для робітників IT сфери чи рядових користувачів. Важливо, щоб ця система була безкоштовною і не вимагала платної підписки, щоб усі охочі мали змогу скористатися необхідними функціями без додаткових витрат.

Крім того, така система віддаленого доступу повинна бути гнучкою і мати велику кількість функцій, корисних як для фахівців IT сфери, так і для звичайних користувачів. Наприклад, вона може містити можливості для керування файлами і папками, віддаленого виконання програм, передачі файлів, доступу до мережевих ресурсів і багато іншого.

Створення такої безкоштовної і гнучкої системи віддаленого доступу може вирішити проблему доступності і полегшити роботу користувачів, дозволяючи їм працювати з будь-якого місця без необхідності великих витрат на підписки.

Така система віддаленого доступу має потенціал стати цінним інструментом для бізнесу та приватних користувачів. Для бізнесу це означає зниження витрат на програмне забезпечення та підписки, що може мати позитивний вплив на бюджет компанії. Крім того, здатність працювати

віддалено дозволяє бізнесу привернути талановитих співробітників з усього світу, не обмежуючись географічними межами.

Для приватних користувачів система віддаленого доступу відкриває широкі можливості для особистого використання. Вона дозволяє отримати доступ до особистих файлів і даних з будь-якого пристрою, а також дозволяє запускати програми та виконувати завдання на віддаленому комп’ютері, що дає гнучкість і зручність у повсякденному житті.

1 ТЕОРЕТИЧНА ЧАСТИНА

1.1 Аналіз застосунків

Застосування систем віддаленого доступу є широким і активно розвивається напрямом в сучасній інформаційній технології. У цьому розділі проводиться аналіз різних застосунків систем віддаленого доступу, що використовуються у різних сферах [1].

Перш за все, розглядаються застосування систем віддаленого доступу в бізнес-середовищі. Такі системи є незамінним інструментом для віддаленого керування комп’ютерами, серверами та мережевими ресурсами. Вони дозволяють працівникам здійснювати роботу з будь-якого місця, забезпечуючи гнучкість та ефективність бізнес-процесів.

Крім того, системи віддаленого доступу широко використовуються в сфері технічної підтримки та обслуговування. Технічні спеціалісти можуть віддалено отримувати доступ до комп’ютерів та інших пристроїв, щоб надати допомогу вирішенню проблем, встановленню програмного забезпечення чи налаштуванню систем.

Також розглядаються застосування систем віддаленого доступу у сфері освіти та дистанційного навчання. Вони дозволяють викладачам та учням спілкуватися, обмінюватися матеріалами та відкрити доступ до віддалених ресурсів навчання незалежно від місцезнаходження.

Крім того, системи віддаленого доступу знайшли застосування у сфері медицини, дозволяючи лікарям віддалено консультувати пацієнтів, моніторити їх стан та передавати медичні дані.

Аналіз різноманітних застосунків систем віддаленого доступу виявляє широкий спектр їх застосування в різних галузях та сферах діяльності. Деякі з них включають:

TeamViewer: є одним з найпопулярніших прикладів систем віддаленого доступу. Він надає можливість віддалено підключатися до комп'ютера чи пристрою, надавати допомогу або виконувати задачі на відстані. TeamViewer працює через Інтернет і дозволяє передавати екран, керувати мишею та клавіатурою, передавати файли та забезпечувати безпеку підключення.

AnyDesk: інше популярне прикладання системи віддаленого доступу. Воно пропонує високу швидкість передачі даних та низьку затримку. AnyDesk підтримує кросплатформенність, що дозволяє використовувати його на різних операційних системах, включаючи Windows, macOS, Linux, Android та iOS.

Remote Desktop Protocol (RDP): є протоколом, що дозволяє віддалене підключення до комп'ютера під управлінням операційної системи Windows. Він забезпечує можливість віддаленого доступу до робочого столу, дозволяє передавати файли та виконувати різні завдання. RDP є стандартним інструментом для віддаленого доступу в операційних системах Windows [7].

VNC (Virtual Network Computing): є іншим популярним протоколом для віддаленого доступу. Він дозволяє віддалено контролювати робочий стіл комп'ютера та передавати його зображення на віддалений пристрій. VNC також підтримує кросплатформенність і доступний на різних операційних системах [1].

Chrome Remote Desktop: Це безкоштовний інструмент для віддаленого доступу, розроблений Google. Він дозволяє встановлювати з'єднання з віддаленими комп'ютерами через браузер Google Chrome. Chrome Remote Desktop простий у використанні і надає базові функції для керування робочим столом і передачі файлів.

Splashtop: Splashtop є ще одним варіантом для віддаленого доступу до комп'ютерів і мобільних пристроїв. Він підтримує передачу високоякісного відео, забезпечуючи плавну роботу на віддаленому робочому столі. Splashtop також має додаткові функції, такі як спільна робота, потокове відтворення мультимедіа і безпека з'єднання.

LogMeIn: LogMeIn є комерційним рішенням для віддаленого доступу, яке надає широкі можливості для керування віддаленими комп’ютерами і системами. Він підтримує безпеку з’єднання, передачу файлів, друку, роботу з декількома моніторами та інші функції.

Ці приклади демонструють різні аспекти систем віддаленого доступу та їх застосування в різних галузях. Однак, варто враховувати, що на ринку існує багато інших рішень та прикладів, які можуть мати свої особливості та переваги залежно від конкретних потреб та вимог користувача.

1.2 Технічне завдання

1.2.1 Аналіз вимог

Функціональне призначення системи – реалізувати та впровадити систему віддаленого доступу для звичайних користувачів або для користувачів IT компаній [4].

Експлуатаційне призначення системи: система може експлуатуватися одним чи більше адміністраторами систем.

В системі мають працювати 2 ролі: адміністратор та віддалений користувач.

У системі роль «віддалений користувач» має можливості заборонити або дозволити отримати адміністратору доступ до свого ПК.

Система має надавати адміністраторам такі можливості:

- доступ до веб-камери віддаленого користувача;
- доступ до файлової системи комп’ютера;
- доступ до комп’ютера віддаленого користувача;
- доступ до консолі операційної системи.

Система повинна бути гнучкою та мати базові можливості коригування інтерфейсу програми, мати зрозумілий та простий інтерфейс для звичайних користувачів;

Система повинна працювати на системі Microsoft Windows 10.

Вимоги до безпеки:

- система не повинна надавати доступ до комп’ютера адміністратора;
- система не повинна надавати доступ до даних комп’ютера адміністратора.

1.2.2 Опис системи

Предметною областю є розробка системи віддаленого доступу. Дана система повинна надати адміністраторам можливість контролю над комп’ютером віддаленого користувача.

Процес взаємодії адміністратора та віддаленого користувача з системою проходить наступним чином.

Адміністратор повинен в системі ініціювати компіляцію .exe файлу, який дозволить підключити користувача до системи віддаленого доступу та передати файл.

Після отримання та запуску .exe файлу користувач зможе побачити у диспетчері завдань декілька процесів які символізують що системі вдалося підключитися.

Адміністратор побачить у інтерфейсі програми основні дані про комп’ютер користувача та отримує доступ до інтерфейсу взаємодії з його комп’ютером.

Процес підключення системи та користувача здійснюється за допомогою системного компоненту Windows discord_game_sdk.dll.

Окрім основних функцій система надає адміністраторам можливість керувати обліковими записами користувачів та швидко зв’язувати їх з новими адміністраторами.

1.2.3 Вимоги до інтерфейсу користувача

Зручність та простота використання – це якісна оцінка застосунку для користувачів як і в IT сфері, так і для звичайних користувачів сайту системи.

При вході на застосунок можна буде побачити на екрані наступний інтерфейс (див. рис. 1.1).

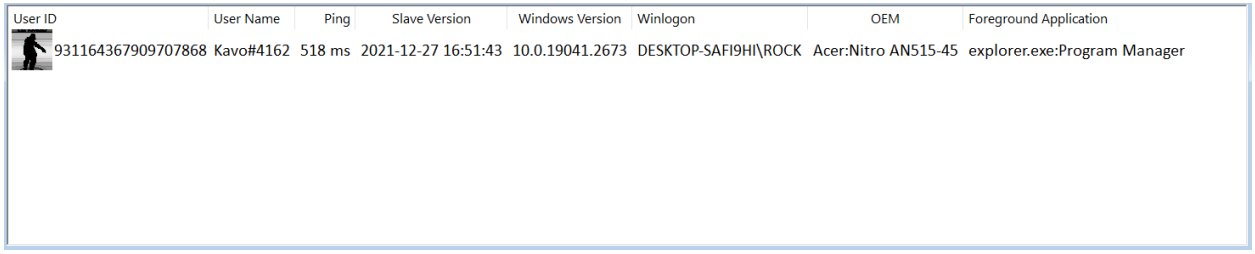

Рисунок 1.1 – Головний інтерфейс застосунку

При роботі із інтерфейсом взаємодії можна буде побачити на екрані наступне (див. рис. 1.2).

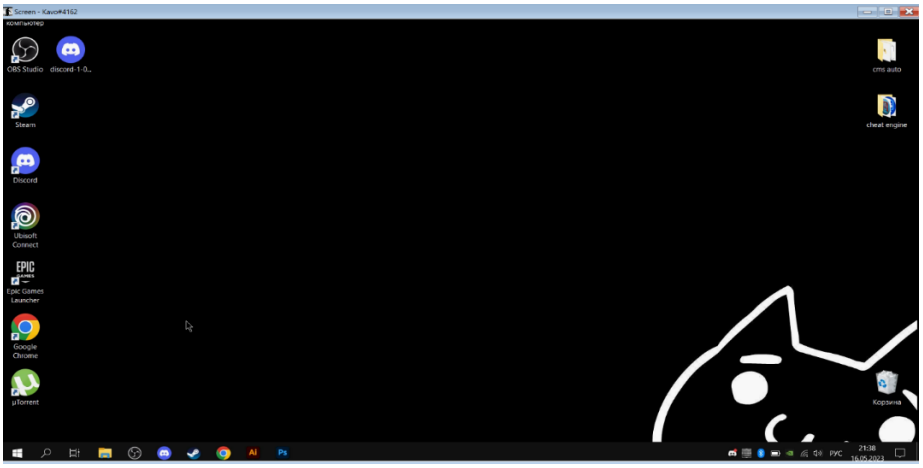

Рисунок 1.2 – Інтерфейс взаємодії з користувачем

При роботі із файловою системою користувача можна буде побачити наступний інтерфейс (див. рис. 1.3).

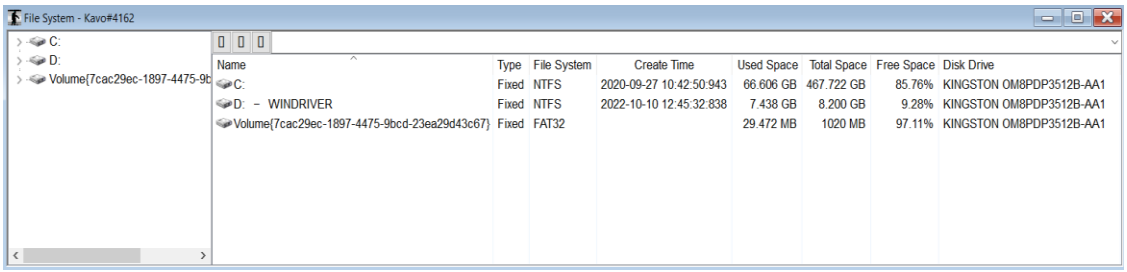

Рисунок 1.3 – Інтерфейс файлової системи

При роботі з консоллю операційної системи можна буде побачити наступний інтерфейс (див. рис. 1.4).

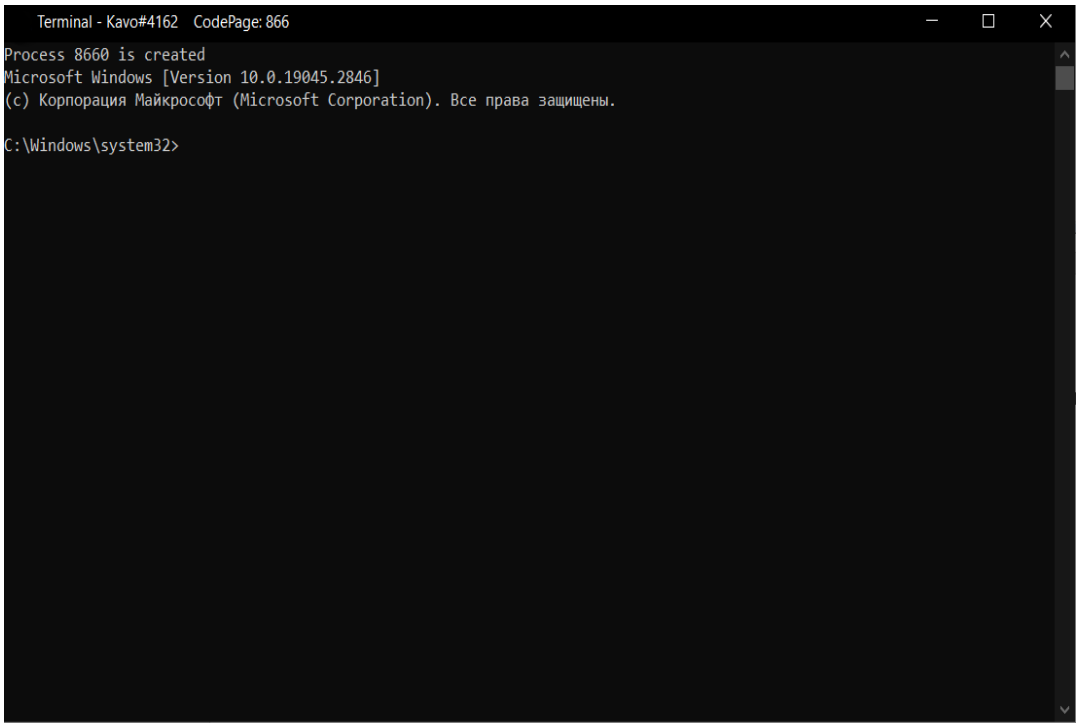

Рисунок 1.4 – Інтерфейс консолі операційної системи

2 ПРАКТИЧНА ЧАСТИНА

2.1 Проєктування системи

Всі методи проєктування систем віддаленого доступу ґрунтуються на концепції створення моделей систем, які можна зобразити графічно, і використовують ці моделі як специфікацію та структуру системи віддаленого доступу [5].

Відобразимо основні цілі та функціональні можливості системи. Для створенная специфікацій та діаграм використаємо хмарний додаток https://www.drawio.com/.

Використаємо діаграму варіантів (Use case diagram) для відображення основних варіантів використання системи віддаленого доступу (рис. 2.1).

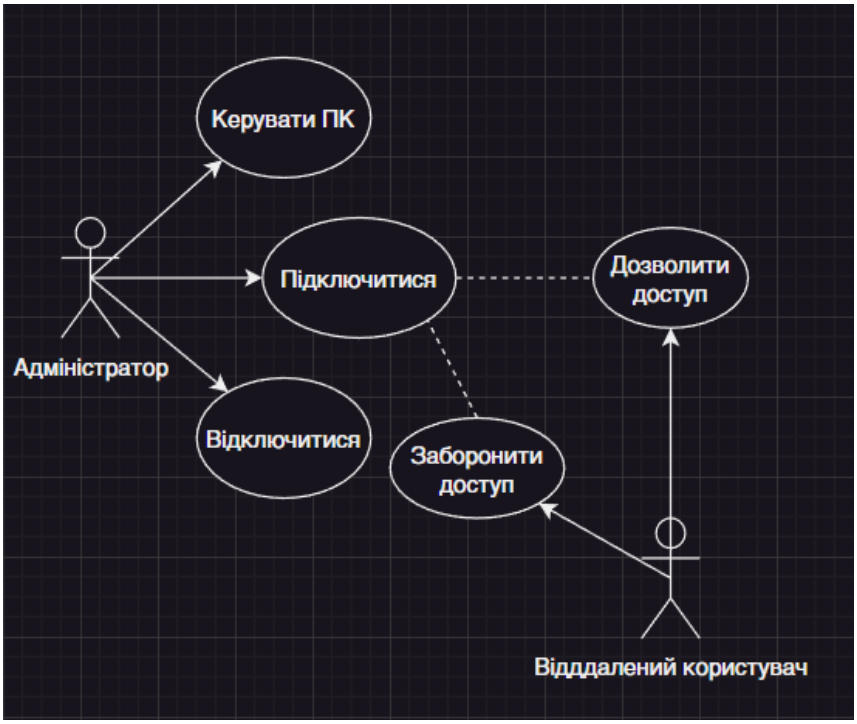

Рисунок 2.1 – Діаграма варіантів

Додаємо опис до варіантів використання (див. рис. 2.1):

1) варіант «Керувати ПК» дає можливість адміністратору керувати ПК користувачем та використовувати функції зазначені у вимогах;
2) варіант «Підключитися» дає можливість адміністратору підключитись до ПК користувача;
3) варіант «Відключитисяя» дає можливість адміністратору відключитись від ПК користувача;
4) варіант «Дозволити доступ» дає можливість користувачу дозволити адміністратору отримати доступ до свого ПК;
5) варіант «Заборонити доступ» дає можливість користувачу заборонити адміністратору отримати доступ до свого ПК.

Таке просте графічне зображення демонструє замовнику основні планові цілі розробки системи віддаленого доступу, щоб усі учасники розробки мали одне спільне уявлення про функціонал.

На прикладі діаграми послідовності для варіанту використання «Керування ПК» (див. рис. 2.2), показано успішний хід подій. Таким чином, будь-який варіант використання може бути відображений у подібній схемі діаграми послідовності.

На цій діаграмі присутні такі етапи взаємодії:

1) адміністратор підключається до віддаленого ПК;
2) адміністратор надсилає запит на керування ПК;
3) віддалений Користувач надсилає відповідь із дозволом керування;
4) адміністратор виконує управління ПК;
5) віддалений Користувач надсилає відповідь із результатом керування;
6) адміністратор відключається від віддаленого ПК.

Ця діаграма послідовності показує взаємодію між Адміністратором і Віддаленим користувачем під час керування ПК. У кожному кроці показано повідомлення, що передаються між акторами.

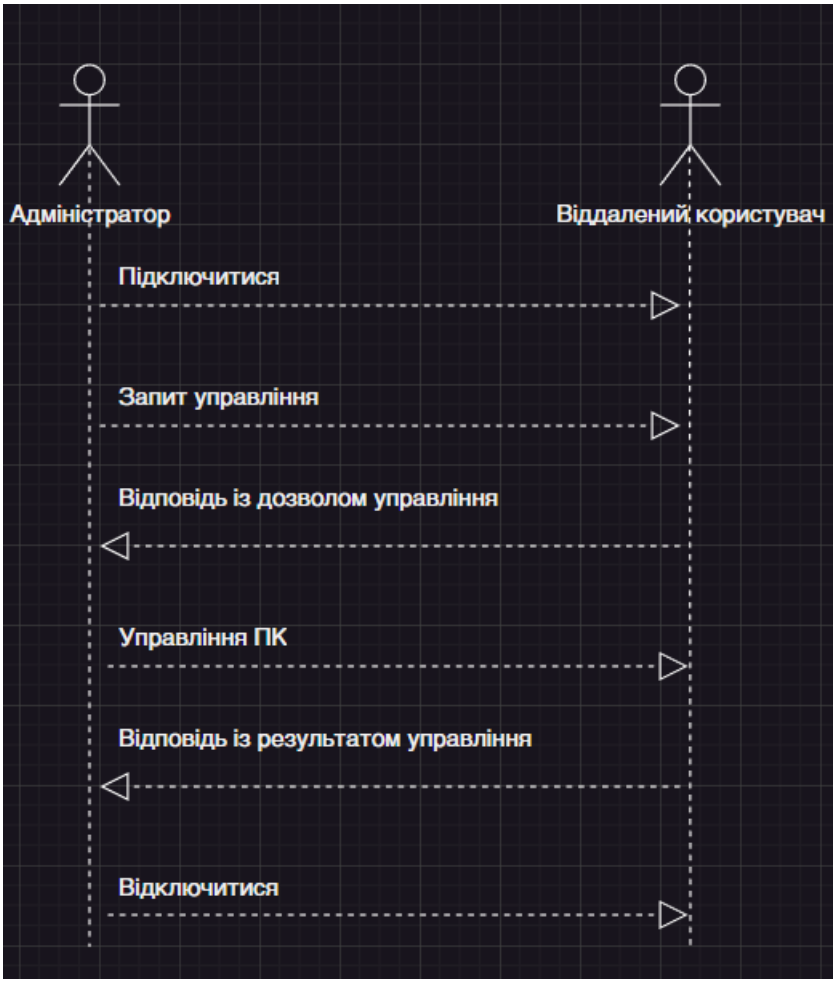

Рисунок 2.2 – Діаграма послідовності для варіанту використання «Керування ПК»

Послідовність кроків відображає порядок взаємодії (див. рис. 2.3).

На цій діаграмі зображені такі класи:

1) адміністратор: людина, яка здійснює віддалене підключення до комп’ютера користувача та має можливість керувати ним;

2) клієнт: програмне забезпечення, встановлене на комп’ютері Адміністратора, яке дозволяє керувати віддаленим комп’ютером;

3) сервер: центральний сервер, який керує та обробляє підключення між Адміністратором та Віддаленим користувачем;

4) віддалений користувач: комп’ютерний користувач, до якого здійснюється віддалений доступ.

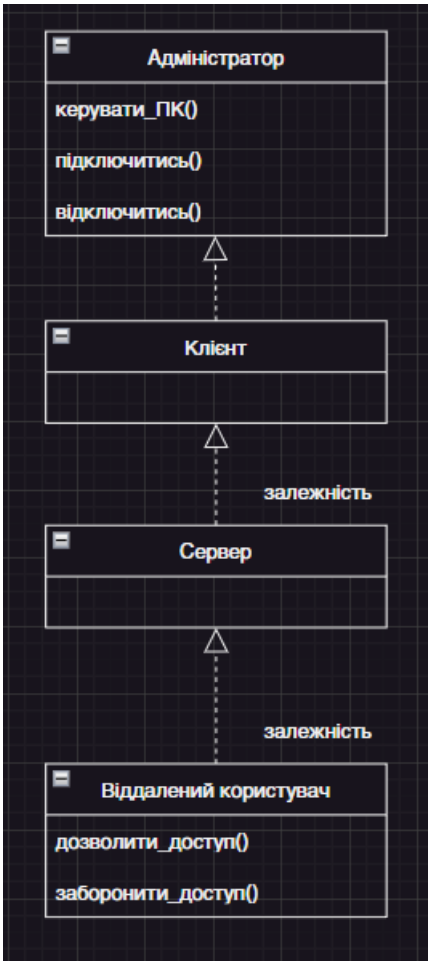

Рисунок 2.3 – Діаграма структури класів

На діаграмі вказано методи (операції) для класу Адміністратора, такі як підключитись(), відключитись() та керувати_ПК(). Залежності показано стрілками вниз, що означає, що клієнт та сервер залежать від класу Адміністратора. Для класу Віддаленого користувача вказано операції, такі як дозволити_доступ(), заборонити_доступ().

Варіант використання «Керувати ПК» дозволяє Адміністратору мати віддалений доступ до комп’ютера користувача через сервер. Під час виконання цього варіанту використовуються наступні компоненти системи:

– компоненти Front-end, які знаходяться на стороні клієнта (Адміністратора) і відповідають за його роботу в системі віддаленого доступу;
– компоненти Back-end, які розташовані на сервері і відповідають за обробку запитів, керування ПК.

Підключаючись до ПК користувача, Адміністратор взаємодіє з інтерфейсом Front-end, який забезпечує зручний спосіб взаємодії з системою. Компоненти Front-end передають необхідну інформацію від Адміністратора до компонентів Back-end на сервері. Компоненти Back-end обробляють цю інформацію та виконують необхідні операції, такі як керування ПК. Результати керування та стан ПК передаються назад на компоненти Front-end для відображення Адміністратору.

Таким чином, варіант використання «Керувати ПК» передбачає взаємодію між компонентами Front-end на стороні клієнта та компонентами Back-end на сервері, що дозволяє Адміністратору віддалено керувати ПК користувача через сервер системи віддаленого доступу.

2.2 Огляд інструментів обробки

Специфікація інструментів для віддаленого керування та розробки програмного забезпечення викликає значний інтерес у галузі інформаційних технологій. У рамках даної дипломної роботи проводиться детальний аналіз таких інструментів, як Remote Desktop Protocol (RDP), VNC (Virtual Network Computing), SSH (Secure Shell), Visual Studio та Assembler. Кожен з цих інструментів має свої особливості та застосовується у різних сферах інформаційних технологій [9].

Протокол віддаленого робочого столу (RDP) є пропрієтарним протоколом, розробленим компанією Microsoft, який дозволяє користувачам віддалено керувати комп’ютерами або віртуальними машинами через мережу. Цей протокол вбудований у операційні системи Microsoft, такі як Windows, що робить його широко доступним для використання. RDP дозволяє встановлювати з’єднання з віддаленим комп’ютером і взаємодіяти з ним, використовуючи віддалений робочий стіл. Ви можете переглядати екран віддаленого комп’ютера, керувати мишею та клавіатурою, запускати

програми та виконувати різні операції, ніби ви фізично знаходитесь перед комп’ютером.

Використовуючи RDP, ви можете встановлювати з’єднання з віддаленим комп’ютером і взаємодіяти з ним, використовуючи віддалений робочий стіл. Один з головних плюсів цього протоколу полягає в тому, що він забезпечує передачу повного образу робочого столу на віддалений комп’ютер, що дозволяє вам бачити і керувати всіма аспектами робочого середовища.

За допомогою RDP можна переглядати екран віддаленого комп’ютера, керувати мишею та клавіатурою, запускати програми та виконувати різні операції, ніби ви фізично знаходитесь перед комп’ютером. Ви можете передавати файли між локальною та віддаленою системами, скопіювати й вставити текст, налаштовувати налаштування екрана та звуку.

Окрім цього, RDP забезпечує безпеку з’єднання, шифруючи передані дані та надаючи можливість аутентифікації для перевірки прав доступу користувача. Такі заходи захисту роблять його надійним засобом для віддаленого доступу та роботи з комп’ютерами в безпечних умовах.

Протокол RDP широко використовується в організаціях, де потрібна віддалена адміністрація комп’ютерів, технічна підтримка, віддалена робота та спільна робота над проєктами. Він дозволяє ефективно використовувати ресурси комп’ютера, підвищує продуктивність та спрощує взаємодію між користувачами на віддалених машинах.

Загалом, RDP є потужним і надійним протоколом, який дозволяє зручний та безпечний віддалений доступ до комп’ютерів і віртуальних машин, спрощуючи роботу та спільне виконання завдань віддалено [2].

VNC (Virtual Network Computing) є вільним програмним забезпеченням, яке надає можливість віддаленого доступу до комп’ютерів або серверів через мережу. Ця технологія заснована на протоколі VNC, який дозволяє забезпечити віддалений контроль і спільний доступ до робочого столу іншого комп’ютера.

Основною перевагою VNC є можливість керування віддаленим комп’ютером, його екраном, мишею та клавіатурою. Ви можете виконувати різні дії на віддаленому комп’ютері, незалежно від його фізичного розташування. Наприклад, ви можете віддалено встановлювати програми, налаштовувати систему, відкривати та редагувати файли, виконувати діагностику проблем та надавати технічну підтримку.

Програми VNC доступні для різних операційних систем, таких як Windows, macOS, Linux і багато інших. Це дозволяє використовувати VNC на різних пристроях, включаючи персональні комп’ютери, ноутбуки, планшети та смартфони. Наявність VNC-клієнта та VNC-сервера на різних платформах робить цю технологію універсальною та зручною для використання в різних сценаріях.

Крім того, VNC також підтримує передачу файлів між віддаленими комп’ютерами, що дає змогу легко обмінюватися даними без необхідності використовувати додаткові засоби передачі. Ви можете передавати файли в обидва напрямки між віддаленим та локальним комп’ютерами безпосередньо з програми VNC.

Загальнодоступність і універсальність програм VNC роблять їх популярними серед користувачів, які потребують віддаленого доступу та керування комп’ютерами. Незалежно від вашого місця перебування, ви можете з легкістю підключитися до віддаленого комп’ютера і працювати з ним, максимально забезпечуючи зручність та ефективність використання [8].

SSH (Secure Shell) є криптографічним мережевим протоколом, який забезпечує безпечний віддалений доступ до комп’ютерів і передачу даних через незахищену мережу. Використовуючи SSH, ви можете встановлювати зашифроване з’єднання з віддаленим сервером і виконувати команди на ньому. SSH забезпечує шифрування даних, аутентифікацію та захист від атак, що робить його популярним варіантом для віддаленого керування серверами і роботою з віддаленими системами [3].

Одна з головних переваг SSH полягає в тому, що він забезпечує шифрування даних під час їх передачі по мережі. Це означає, що навіть якщо хтось зловмисник здобуде доступ до переданих даних, він не зможе розшифрувати їх без відповідного ключа. Таким чином, SSH забезпечує конфіденційність інформації під час її трансляції по мережі.

Крім шифрування, SSH також надає механізм аутентифікації, що дозволяє перевірити, що користувач, який намагається отримати доступ до віддаленого сервера, є дійсним і має відповідні права доступу. Це забезпечує безпеку системи, оскільки запобігає несанкціонованому доступу до важливих ресурсів.

SSH також володіє механізмами захисту від різних видів атак, включаючи атаки перехоплення, введення підроблених даних і перехоплення паролів. Він використовує різні криптографічні протоколи і методи, що робить його надійним і ефективним засобом для віддаленого керування серверами і роботи з віддаленими системами.

Протокол SSH широко використовується в галузі інформаційної безпеки, веб-розробці, адмініструванні серверів і багатьох інших сферах, де віддалений доступ і захищена передача даних є важливими аспектами. Існують різні програмні реалізації SSH для різних платформ, включаючи відкриті і безкоштовні варіанти, що робить його доступним і популярним серед користувачів по всьому світу.

Visual Studio є інтегрованою середою розробки (IDE), розробленою компанією Microsoft. Це потужне програмне забезпечення, призначене для розробки програмного забезпечення на різних платформах, таких як Windows, macOS і Linux. Visual Studio має різноманітні інструменти і функції, які допомагають розробникам писати, налагоджувати і тестувати код, створювати графічні інтерфейси користувача та виконувати інші задачі, пов’язані з розробкою програмного забезпечення [11].

Assembler є мовою програмування низького рівня, яка використовується для написання програм для процесорів і мікроконтролерів.

Вона дозволяє розробникам працювати з низькорівневими операціями, такими як робота з реєстрами, пам’яттю і регістрами процесора. Assembler забезпечує прямий контроль над апаратними можливостями системи і дає можливість оптимізувати код для підвищення продуктивності. Використання Assembler вимагає глибокого розуміння апаратної архітектури і деталей процесора [6].

Ці інструменти мають свої особливості і застосовуються для різних цілей, таких як віддалене керування комп’ютерами, розробка програмного забезпечення та робота з апаратними ресурсами. Вони допомагають розширити можливості і забезпечити ефективну роботу з комп’ютерами та серверами.

2.3 Реалізація системи віддаленого доступу

2.3.1 Реалізація роботи консолі операційної системи

Задля забезпечення ефективної роботи програми віддаленого доступу та взаємодії з консоллю іншого користувача, важливо врахувати різноманітні аспекти цього процесу. Одним із ключових завдань є забезпечення безпеки та конфіденційності передачі даних між віддаленим сервером та локальною консоллю.

Для забезпечення безпеки передачі даних можна використовувати різні методи і протоколи шифрування, які дозволяють захистити інформацію від несанкціонованого доступу. Наприклад, використання протоколу SSH (Secure Shell) забезпечує шифрування трафіку та аутентифікацію користувача, що забезпечує безпеку під час взаємодії з консоллю.

Окрім того, для зручності користувача важливо реалізувати зручний і інтуїтивно зрозумілий інтерфейс консолі, який дозволяє здійснювати різні операції та взаємодіяти з віддаленим сервером. Наприклад, можна розробити

командний рядок з розширеними можливостями, що дозволяє виконувати команди, переглядати статус системи, керувати процесами та іншими функціями.

Для покращення продуктивності та швидкості роботи програми віддаленого доступу, можна використовувати різні оптимізації та кешування даних. Наприклад, можна зберігати локальну копію файлів або результатів попередніх операцій, щоб уникнути зайвого мережевого трафіку та забезпечити швидкий доступ до інформації.

Крім того, важливо передбачити можливі ситуації з низькою якістю зв’язку або відключенням, і розробити механізми для перез’єднання та відновлення роботи після таких подій. Наприклад, програма може автоматично спробувати перепідключитися до віддаленого сервера після втрати зв’язку, зберігаючи стан та контекст роботи.

Взаємодія з консоллю іншого користувача віддалено відкриває безліч можливостей і переваг. Наприклад, можна здійснювати адміністративні функції на віддаленому сервері, дистанційно керувати системними налаштуваннями, виконувати скрипти та автоматизовані завдання. Це особливо корисно для системних адміністраторів та команд розробників, які потребують швидкого та зручного доступу до віддалених серверів.

Реалізація взаємодії роботи програми віддаленого доступу з консоллю іншого користувача має великий потенціал і може бути використана в різних сферах, починаючи від системного адміністрування та закінчуючи розробкою програмного забезпечення. Вона дозволяє ефективно керувати віддаленими ресурсами, спрощує процеси взаємодії та забезпечує більш гнучкий та продуктивний робочий процес.

У фрагменті коду демонструється реалізація взаємодії з консоллю операційної системи в системі віддаленого доступу (див. Додаток А).

Код містить набір підключених бібліотек та функцій, які дозволяють отримувати введені дані з консолі та передавати їх до віддаленого пристрою.

У даному випадку використовується API операційної системи Windows для взаємодії з консоллю. Код забезпечує зчитування введених даних з консолі, конвертацію тексту в різні кодування та відправку цих даних до віддаленого пристрою. Він також містить обробники подій консолі, які реагують на введення користувача та управління процесами.

Надана реалізація дозволяє ефективно працювати з консоллю операційної системи під час віддаленого доступу, що використовується, наприклад, в системах керування віддаленими пристроями або віддаленими серверами.

Даний фрагмент коду реалізує взаємодію з консоллю операційної системи в системі віддаленого доступу. Він містить функції та процедури, які дозволяють отримувати ввід від користувача через консоль та передавати цей ввід на віддалений сервер.

Основні елементи цього коду наступні:

1) підключення необхідних заголовочних файлів, таких як stddef.h, stdint.h, stdio.h, stdlib.h, time.h, math.h, ntdef.h, winternl.h, windows.h, windowsx.h, commctrl.h, а також користувацьких заголовочних файлів misc.h, guihalp.h, cmdbox.h;
2) визначення глобальних змінних, таких як HWND MsgWindow, int64_t lobby_id, user_id, int32_t hWritePipe, RemotePID, unsigned ExpectedTimestamp, LastTimestamp, wchar_t* username, wchar_t s[4096], bool FirstInstance, unsigned LastCodePage;
3) оголошення функції ReadInputThreadProc, яка відповідає за зчитування введення користувача з консолі;
4) оголошення функції ConsoleCtrlHandler, яка відповідає за обробку сигналів управління консоллю, таких як CTRL+C або CTRL+BREAK;
5) оголошення функції MessageWindowProc, яка відповідає за обробку повідомлень вікна, що використовується для передачі даних між процесами;

6) функція ShowCMDBox, яка створює консольне вікно та запускає процес віддаленого доступу.

Цей фрагмент коду реалізує деякі функціональні можливості для взаємодії з консоллю операційної системи в контексті системи віддаленого доступу.

Для більш глибокого розуміння роботи функцій є докладніші описи кожної із згаданих у коді:

1) функція ReadInputThreadProc: ця функція відповідає за зчитування введення користувача з консолі; вона запускається у власному потоці виконання і працює у фоновому режимі, без перешкоджання основному потоці програми; функція використовує вбудовані функції та методи API операційної системи для отримання введених даних з консолі та їх обробки;
2) функція ConsoleCtrlHandler: ця функція відповідає за обробку сигналів управління консоллю, таких як CTRL+C або CTRL+BREAK ; вона реєструється як обробник подій консолі і викликається в разі отримання відповідних сигналів; функція дозволяє виконати певні дії або встановити певні стани програми в залежності від отриманих сигналів;
3) функція MessageWindowProc: ця функція відповідає за обробку повідомлень вікна, що використовується для передачі даних між процесами; вона викликається системою оперативно при отриманні повідомлень для вікна і може містити логіку обробки цих повідомлень; функція дозволяє передавати дані між процесами, використовуючи механізми комунікації операційної системи;
4) функція ShowCMDBox: ця функція створює консольне вікно та запускає процес віддаленого доступу; вона використовує API операційної системи для створення вікна та налаштування його властивостей; після створення вікна функція запускає процес

віддаленого доступу, використовуючи вбудовані функції та методи для цього.

Ці функції мають важливе значення для забезпечення взаємодії з консоллю операційної системи в системі віддаленого доступу. Вони допомагають отримувати введені дані від користувача, обробляти сигнали управління консоллю, обмінюватися даними між процесами та керувати процесом віддаленого доступу.

Для захисту даних під час взаємодії з консоллю використовується протокол SSH (Secure Shell). SSH є криптографічним протоколом, який забезпечує безпечний обмін даними між віддаленим сервером та локальною консоллю.

Протокол SSH забезпечує шифрування трафіку, аутентифікацію користувача і захист від атак перехоплення даних. Під час взаємодії з консоллю, SSH використовує симетричне шифрування для захисту передачі даних, що означає, що дані, передані між віддаленим сервером і локальною консоллю, шифруються з використанням спільного ключа.

Крім шифрування, SSH використовує механізми аутентифікації для перевірки ідентичності користувача перед доступом до віддаленого сервера. Зазвичай, це здійснюється за допомогою пари ключів, яка складається з приватного і публічного ключів. Приватний ключ знаходиться на локальній системі, а публічний ключ розповсюджується на віддалений сервер. Під час аутентифікації, SSH використовує ці ключі для перевірки користувача і дозволяє доступ тільки при відповідності ключів.

Застосування протоколу SSH забезпечує високий рівень безпеки під час взаємодії з консоллю, дозволяючи захистити дані від несанкціонованого доступу та перехоплення. Це робить SSH незамінним інструментом для віддаленого доступу та адміністрування систем, забезпечуючи конфіденційність та безпеку передачі даних.

2.3.2 Реалізація доступу до екрану та камери користувача

При розробці систем віддаленого доступу іноді доцільно використовувати додаткові функціональні можливості для поліпшення користувальницького досвіду. Однією з таких можливостей може бути функція контролю за екраном віддаленого користувача. Ця функція дозволяє переглядати та контролювати віддалений екран в режимі реального часу. Користувач може бачити, що відбувається на екрані віддаленого комп’ютера і виконувати необхідні дії.

Крім того, як інноваційна можливість, може бути розроблена функція доступу до веб-камери користувача. Ця функція дозволяє віддаленому користувачу отримувати доступ до веб-камери свого комп’ютера і передавати відео зображення в режимі реального часу. Це може бути корисно, наприклад, при відеоконференціях або спілкуванні з віддаленими співробітниками чи клієнтами.

Застосування віддаленого доступу має широкий спектр використання, включаючи технічну підтримку, віддалену роботу, спільне використання ресурсів і багато іншого. Функції контролю за екраном та доступу до веб-камери можуть значно покращити продуктивність та зручність використання систем віддаленого доступу для різних сценаріїв застосування.

У фрагменті коду представлено підключення необхідних файлів заголовків та визначення, необхідних для роботи вікна перегляду екрану (див. додаток Б).

Перед описом функцій і макроозначень, включених у код, важливо зрозуміти, що це лише фрагмент і, ймовірно, є частиною більшої програми або проєкту. Код містить декілька директив #include, які підключають необхідні бібліотеки та заголовочні файли для роботи з вікнами, графічними елементами і засобами зображення.

Також у коді використовуються деякі визначення макросів (#define), які призначені для надання ідентифікаторів для певних значень, таких як

ідентифікатори меню (IDM_) і базові індекси пристроїв (IDM_DEVBASE і IDM_RESBASE).

Важливим аспектом цього фрагменту є наявність функцій, які оновлюють вікно перегляду екрану (RefreshScreen) та вікно камери (RefreshCamera). Ці функції приймають параметри, такі як вказівник на вік.

Даний фрагмент коду включає декілька заголовочних, а також визначає константи та функції для роботи з вікном перегляду екрану. Нижче наведено опис деяких функцій:

1) validateScreenViewerWindow: перевіряє, чи є вікно дійсним вікном перегляду екрану, шляхом перевірки його параметрів;
2) refreshScreen: оновлює зображення екрану у вікні перегляду, параметр hwnd вказує на вікно, restart вказує, чи потрібно рестартувати зображення з початку, а channel вказує номер каналу для передачі повідомлення, функція складає повідомлення про оновлення екрану, встановлює властивості повідомлення і відправляє його головному вікну програми;
3) refreshCamera: оновлює зображення камери у вікні перегляду. Параметри hwnd, restart і channel мають ті ж самі значення, що й у функції RefreshScreen, функція складає повідомлення про оновлення камери, встановлює властивості повідомлення і відправляє його головному вікну програми;
4) maxScreenViewSize: максимізує розмір вікна перегляду екрану з урахуванням обмежень екрану, встановлює розмір і позицію вікна залежно від розміру екрану і параметрів вікна;
5) resetScreenViewSize: скидає розмір вікна перегляду екрану на задані значення cx і cy, ця функція створює новий буфер даних для зображення екрану з врахуванням нового розміру вікна;
6) printTextToBitmap: виводить текст на зображенні екрану у вікні перегляду, функція отримує рядок символів s і його довжину n, а потім рендерить текст на бітову карту (bitmap) зображення екрану.

Весь код видається фрагментом програми для перегляду екрану або камери віддаленого комп’ютера. Цей фрагмент містить функції для оновлення зображення, масштабування вікна та інших взаємодій з вікном перегляду.

3 ТЕСТУВАННЯ

3.1 Тестування системи віддаленого доступу

Для проведення тестування системи віддаленого доступу потрібно виконати наступні кроки, що описані нижче.

Реєстрація користувача:

– завантажити .exe-файл, який передбачено для реєстрації користувача;
– виконати файл для встановлення з'єднання з системою віддаленого доступу.

Ролі користувача:

– перевірити наявність двох ролей: «віддалений користувач» і «адміністратор»;
– перевірити функціональність кожної ролі згідно вимог системи.

Перевірка базових функцій контролю екрана користувача:

– отримати віддалений доступ до комп'ютера користувача;
– перевірити можливість відображення екрана користувача на локальному пристрої;
– перевірити можливість керування мишею та клавіатурою віддаленого комп'ютера;
– перевірити передачу звуку та відео з віддаленого комп'ютера.

Взаємодія з консоллю операційної системи:

– виконати команди на консолі віддаленого комп'ютера з допомогою системи віддаленого доступу;
– перевірити відображення виведеної інформації та результатів команд на локальному пристрої.

Перевірка інших функцій:

– перевірити можливість передачі файлів між віддаленими комп'ютерами;

– перевірити можливість встановлення спеціальних налаштувань, таких як роздільна здатність екрана, на віддаленому комп’ютері.

Роботу SSH протоколу в системі віддаленого доступу було успішно протестовано з використанням наступних кроків:

– перевірено: наявність встановленого SSH сервера на віддаленому комп’ютері;
– здійснено підключення до віддаленого сервера за допомогою SSH-клієнта, використовуючи команду ssh username@host, де username – ім’я користувача, а host – IP-адреса або доменне ім’я віддаленого сервера;
– перевірено: можливість аутентифікації шляхом введення правильного пароля або за допомогою ключа SSH;
– виконано команди на віддаленому сервері, щоб переконатися, що вони виконуються правильно (було введено різні команди, такі як ls, pwd, mkdir, щоб перевірити здатність SSH протоколу до виконання команд на віддаленому сервері);
– перевірено: можливість передачі файлів між локальною та віддаленою системами за допомогою команди scp, щоб переконатися, що файловий обмін через SSH протокол працює належним чином;
– виконано тест на надійність з’єднання, відключивши мережеве з’єднання під час активного сеансу SSH та перевіривши, що з’єднання автоматично відновлюється після відновлення мережі;
– перевірено: можливість зміни конфігураційних параметрів SSH сервера, таких як порт, дозволені методи аутентифікації та інші налаштування, та переконанося, що зміни застосовуються належним чином;
– виконано тест на безпеку, переконавшись, що SSH протокол використовує шифрування даних та ідентифікує віддалених користувачів і сервери для захисту від можливих атак;

– проведено тестування з використанням різних SSH-клієнтів та SSH-серверів, щоб переконатися, що протокол сумісний з різними програмними рішеннями.

Взаємодію роботи з консоллю операційної системи в системі віддаленого доступу було успішно протестовано з використанням наступних кроків:

– підключенося до віддаленого сервера за допомогою SSH протоколу з використанням відповідних авторизаційних даних;
– виконано різні команди на консолі оперативної системи, такі як ls, pwd, cd, mkdir, для перевірки їх взаємодії з віддаленим сервером;
– перевірено можливість введення тексту та виконання команд безпосередньо в консолі віддаленого сервера;
– виконано скрипти або команди з довшими часовими інтервалами, щоб переконатися, що взаємодія з консоллю залишається активною протягом тривалого часу;
– перевірено можливість перенаправлення введення та виведення, наприклад, запуск команди з перенаправленням виведення в файл або зчитування введення з файлу;
– перевірено можливість взаємодії з текстовими редакторами або іншими консольними програмами, що вимагають введення тексту або взаємодії з користувачем;
– проведено тестування різних команд та функцій консолі, щоб переконатися в їх коректному виконанні та відповідному виведенні результатів;
– перевірено можливість роботи з командами терміналу, такими як перемикання між сесіями, створення терміналу в новому вікні або вкладці, відправка сигналів процесам та інші функції терміналу.

Було проведено перевірку базових функцій контролю екрана користувача в системі віддаленого доступу з виконанням наступних кроків:

- перевірено: можливість підключитися до віддаленого комп’ютера за допомогою SSH протоколу або відповідного програмного забезпечення для віддаленого доступу;
- перевірено: функція, що дозволяє переглядати екран віддаленого користувача, наприклад, використовуючи команду screen або функціонал програми віддаленого доступу;
- перевірено: можливість спостереження за діями користувача на екрані в реальному часі;
- перевірено: можливість керування мишею та клавіатурою віддаленого користувача через систему віддаленого доступу;
- перевірено: різні дії на віддаленому комп’ютері, такі як відкриття програм, переходи між вікнами, редагування файлів, щоб переконатися в коректності відображення цих дій на екрані віддаленого користувача;
- перевірено: можливість передачі файлів між локальною та віддаленою системою через систему віддаленого доступу;
- перевірено: зміни, внесені на віддаленому комп’ютері або сервері, відображаються на екрані в реальному часі;
- проведно тестування з різними роздільними здатностями екрану, щоб переконатися в коректному відображенні вмісту.

Були перевірені наступні ролі користувача в системі віддаленого доступу.

Роль «Віддалений користувач»:

- зареєстровано нового користувача в системі віддаленого доступу;
- виконано процес авторизації для віддаленого користувача;
- перевірено можливість встановлення з’єднання з віддаленим комп’ютером;
- здійснено перегляд та керування екраном віддаленого комп’ютера;
- виконано дії на віддаленому комп’ютері, включаючи взаємодію з консоллю операційної системи.

Роль «Адміністратор»:

– зареєстровано нового користувача з ролями адміністратора;
– виконано процес авторизації для адміністратора;
– перевірено розширені можливості, доступні адміністратору, включаючи керування правами доступу, налаштування системних параметрів та контроль за віддаленими користувачами;
– перевірено можливість встановлення спеціальних налаштувань, таких як роздільна здатність екрана, на віддаленому комп’ютері.

Було проведено тестування взаємодії з файловою системою в системі віддаленого доступу. Наступні кроки були виконані та перевірені:

Передача файлів між локальним та віддаленим комп’ютером:

– обрано файл на локальному комп’ютері та відправлено його на віддалений комп’ютер;
– перевірено, чи файл успішно передався та зберігся на віддаленому комп’ютері;
– обрано файл на віддаленому комп’ютері та відправлено його на локальний комп’ютер;
– перевірено, чи файл успішно передався та зберігся на локальному комп’ютері.

Маніпулювання файлами на віддаленому комп’ютері:

– створено нову папку на віддаленому комп’ютері та перевірено, чи вона правильно відображається у файловій системі;
– скопійовано файли з однієї папки на віддаленому комп’ютері до іншої та перевірено, чи вони правильно скопійовані;
– видалено папку та перевірено, чи вона видалилася з віддаленого комп’ютера.

Налаштування спеціальних параметрів файлової системи:

– змінено роздільну здатність екрана на віддаленому комп’ютері та перевірено, чи зміни відображаються коректно;

– встановлено спеціальні налаштування, пов’язані з роботою з файловою системою, такі як обмеження розміру файлу, права доступу тощо, та перевірено їх відповідність заданим значенням.

ВИСНОВКИ

У результаті теоретичної частини було виявлено, що для розробки системи удаленого доступу до комп’ютера найбільш підходящими методами розробки є використання програмних засобів Visual Studio, Assembler Remote Desktop Protocol (RDP), VNC (Virtual Network Computing) та SSH (Secure Shell).

Для проєктування системи були розроблені специфікації, що дозволить представити цільову систему віддаленого доступу у вигляді таких моделей як: функціональна, компонентна та реалізаційна, побудовані відповідні діаграми.

Розробили технічне завдання для розробки системи віддаленого доступу, в якому описали функціональні вимоги, вимоги до безпеки.

Згідно технічного завдання реалізували систему віддаленого у вигляді клієнт-серверної компонентної архітектури.

Данна архітектура включає програму на основі API та язиків програмування C++ та Assembler.

Реалізували варіанти використання системи для ролі Адміністратор.

Реалізований зв’язок між клієнтом та сервером за допомогою API.

Данна система може слугувати прототипом, шаблоном для створення більш складних функціональних систем, використовуючи принципи – зниження складності через компонентну архітектуру – розбивку складних сутностей чи процесів на прості компоненти.

ПЕРЕЛІК ПОСИЛАНЬ

1. Alasooly H. Evaluation of Some Remote Desktop Protocol (RDP) Services Providers. *Amazon*. URL: https://www.amazon.com/dp/B08NF81FHF (дата звернення: 14.03.2023).
2. Anderson C. Windows Server 2008 R2 Remote Desktop Services Resource Kit. *Amazon*. URL: https://www.amazon.com/dp/B00JDMPDA8 (дата звернення: 17.03.2023).
3. Barrett D. SSH, The Secure Shell: The Definitive Guide: The Definitive Guide. *Amazon*. URL: https://www.amazon.com/SSH-Secure-Shell-Definitive-Guides/dp/0596008953 (дата звернення: 20.03.2023).
4. Coulouris G., Dollimore J., Kindberg T. Distributed Systems: Concepts and Design. *Amazon*. URL: https://www.amazon.com/Distributed-Systems-Concepts-Design-5th/dp/0132143011 (дата звернення: 22.03.2023).
5. Dauti B. Windows Server 2019 Administration Fundamentals: A beginner's guide to managing and administering Windows Server environments, 2nd Edition. *Amazon*. URL: https://www.amazon.com/dp/B07YLXVVHJ (дата звернення: 01.04.2023).
6. Dunne R. Windows 64-bit Assembly Language Programming Quick Start: Intel X86-64, SSE, AVX. *Amazon*. URL: https://www.amazon.com/dp/0970112467 (дата звернення: 05.04.2023).
7. Hanrahan E. Windows Internals: System architecture, processes, threads, memory management, and more, Part 1 (Developer Reference). *Amazon*. URL: https://www.amazon.com/Windows-Internals-Part-architecture-management/dp/0735684189 (дата звернення: 07.04.2023).
8. Kurose J., Ros K. Computer Networking: A Top-Down Approach. *Amazon*. URL: https://www.amazon.com/Computer-Networking-Top-Down-Approach-7th/dp/0133594149 (дата звернення: 11.04.2023).

9. Oram A. Peer-to-Peer: Harnessing the Power of Disruptive Technologies. *Amazon*. URL: https://www.amazon.com/dp/B01C7S0L6A (дата звернення: 14.04.2023).
10. Steen M. V. Distributed Systems: Principles and Paradigms. Pearson Education. *Amazon*. URL: https://www.amazon.com/Distributed-Systems-Principles-Paradigms-2nd/dp/0132392275 (дата звернення: 20.04.2023).
11. Stroustrup B. The C++ Programming Language. *Amazon*. URL: https://www.amazon.com/C-Programming-Language-4th/dp/0321563840 (дата звернення: 25.04.2023).

ДОДАТОК А

Реалізація взаємодії з консоллю операційної системи

```
#include <stddef.h>
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>
#include <ntdef.h>
#include <winternl.h>
#include <windows.h>
#include <windowsx.h>
#include <commctrl.h>
#include "misc.h"
#include "guihalp.h"
#include "cmdbox.h"

static HWND MsgWindow;
static int64_t lobby_id,user_id;static int32_t hWritePipe,RemotePID;static unsigned ExpectedTimestamp,LastTimestamp;
static wchar_t*username,s[4096];static bool FirstInstance;static unsigned LastCodePage;
static DWORD CALLBACK ReadInputThreadProc(void*param)
{
	HANDLE hStdInput=GetStdHandle(STD_INPUT_HANDLE),hStdOutput=GetStdHandle(STD_OUTPUT_HANDLE);
	COPYDATASTRUCT cds;
```

```
cds.dwData=0;
const unsigned wstrbuflen=1024,strbuflen=wstrbuflen*4;DWORD wstrlen,strlenu8;
wchar_t*wstrbuf=(wchar_t*)HeapAlloc(Heap,0,wstrbuflen*sizeof(wchar_t));
if(!wstrbuf)return ERROR_NOT_ENOUGH_MEMORY;
if(cds.lpData=HeapAlloc(Heap,0,sizeof(SENDMSGSTRUCT)+sizeof(uint32_t)*2+strbuflen)){
    PSENDMSGSTRUCT(cds.lpData)->lobby_id=lobby_id;
    PSENDMSGSTRUCT(cds.lpData)->user_id=user_id;
    PSENDMSGSTRUCT(cds.lpData)->channel=1;
    PSENDMSGSTRUCT(cds.lpData)->msg.Type=FirstInstance&&hWritePipe==0&&GetAsyncKeyState(VK_SHIFT)>=0?

(GetPrivateProfileIntW(L"MASTER",L"TerminalRunAs",0,INIFileName)?RDP_MESSAGE::StdIOContinueRunAs:RDP_MESSAGE::StdIOContinue):

GetPrivateProfileIntW(L"MASTER",L"TerminalRunAs",0,INIFileName)?RDP_MESSAGE::StdIORunAs:RDP_MESSAGE::StdIO;
    PSENDMSGSTRUCT(cds.lpData)->msg.u[0]=PtrPad^(UINT_PTR)MsgWindow;
    char*strbuf=PSTR(PSENDMSGSTRUCT(cds.lpData)->msg.u+2);

if(wstrlen=GetPrivateProfileStringW(L"MASTER",L"TerminalAutoRun",0,wstrbuf,wstrbuflen,INIFileName)){

strlenu8=WideCharToMultiByte(CP_UTF8,0,wstrbuf,wstrlen,strbuf,strbuflen,0,0);
```

```
                PSENDMSGSTRUCT(cds.lpData)-
>msgsize=RDP_MESSAGE_HEADER_SIZE+sizeof(int32_t)*2+strlenu8;

        cds.cbData=offsetof(SENDMSGSTRUCT,msg)+PSENDMSGSTRUCT(cds
.lpData)->msgsize;
                PSENDMSGSTRUCT(cds.lpData)->msg.u[1]=hWritePipe;
                PSENDMSGSTRUCT(cds.lpData)-
>msg.Timestamp=ExpectedTimestamp=GetTimestamp();

        SendMessageW(MainWindow,WM_COPYDATA,(WPARAM)MsgWindo
w,(LPARAM)&cds);
            }
            for(;;)
            if(ReadConsoleW(hStdInput,wstrbuf,wstrbuflen,&wstrlen,0)){
                if(wstrlen<2||wstrbuf[wstrlen-2]!=L'\r'||wstrbuf[wstrlen-
1]!=L'\n'){
                    DWORD n;
                    WriteConsoleW(hStdOutput,L"\r\n",2,&n,0);
                }else                   while(wstrlen>0&&(wstrbuf[wstrlen-
1]==L'\r'||wstrbuf[wstrlen-1]==L'\n'))--wstrlen;

        if(wstrlen==0&&hWritePipe==0)wstrlen=GetPrivateProfileStringW(L"MA
STER",L"TerminalAutoRun",0,wstrbuf,wstrbuflen,INIFileName);

        strlenu8=WideCharToMultiByte(CP_UTF8,0,wstrbuf,wstrlen,strbuf,strbufle
n,0,0);
                PSENDMSGSTRUCT(cds.lpData)-
>msgsize=RDP_MESSAGE_HEADER_SIZE+sizeof(int32_t)*2+strlenu8;
```

```
        cds.cbData=offsetof(SENDMSGSTRUCT,msg)+PSENDMSGSTRUCT(cds
.lpData)->msgsize;
                        PSENDMSGSTRUCT(cds.lpData)->msg.u[1]=hWritePipe;
                        if(hWritePipe)PSENDMSGSTRUCT(cds.lpData)-
>msg.Type=GetPrivateProfileIntW(L"MASTER",L"TerminalRunAs",0,INIFileNa
me)?RDP_MESSAGE::StdIORunAs:RDP_MESSAGE::StdIO;
                        PSENDMSGSTRUCT(cds.lpData)-
>msg.Timestamp=ExpectedTimestamp=GetTimestamp();

        SendMessageW(MainWindow,WM_COPYDATA,(WPARAM)MsgWindo
w,(LPARAM)&cds);
                }else{
                        DWORD err=GetLastError(),n=sizeof"ReadConsoleW failed:";
                        wcscpy(wstrbuf,L"ReadConsoleW failed: ");

        n+=FormatMessageW(FORMAT_MESSAGE_FROM_SYSTEM,0,err,0,ws
trbuf+n,wstrbuflen-n-2,0);
                        if(wstrbuf[n-1]!=L'\n')wstrbuf[n++]=L'\r',wstrbuf[n++]=L'\n';
                        WriteConsoleW(hStdOutput,wstrbuf,n,&n,0);
                }
                HeapFree(Heap,0,cds.lpData);
                ExitProcess(0);
        }
        HeapFree(Heap,0,wstrbuf);
        return ERROR_NOT_ENOUGH_MEMORY;
}

static BOOL WINAPI ConsoleCtrlHandler(DWORD dwCtrlType)
{
```

```
	if(dwCtrlType==CTRL_C_EVENT||dwCtrlType==CTRL_BREAK_EVENT||dwCtrlType==CTRL_CLOSE_EVENT){
		if(RemotePID&&MessageBoxW(0,L"Terminate the remote process?",L"Terminal",MB_YESNO|MB_ICONQUESTION)==IDYES){

	allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT)+sizeof(uint32_t))
			COPYDATASTRUCT cds={0,sizeof(SENDMSGSTRUCT)+sizeof(uint32_t),&sm};
			sm.lobby_id=lobby_id;
			sm.user_id=user_id;
			sm.channel=1;

	sm.msgsize=RDP_MESSAGE_HEADER_SIZE+sizeof(int32_t);
			sm.msg.Type=RDP_MESSAGE::EndProcessTree;
			sm.msg.Timestamp=ExpectedTimestamp=GetTimestamp();
			sm.msg.u[0]=RemotePID;

	SendMessageW(MainWindow,WM_COPYDATA,(WPARAM)MsgWindow,(LPARAM)&cds);
		}
		ExitProcess(0);
		return 1;
	}else return 0;
}

static LRESULT CALLBACK MessageWindowProc(HWND hwnd,UINT uMsg,WPARAM wParam,LPARAM lParam)
{
	switch(uMsg){
```

```
            case WM_CREATE:return 0;
            case WM_QUERYENDSESSION:return 1;
            case WM_ENDSESSION:if(wParam)PostQuitMessage(0);return 0;
            case WM_CLOSE:return 0;
            case WM_DESTROY:return 0;
            case WM_COPYDATA:{
                PRDP_MESSAGE
msg=PRDP_MESSAGE(PCOPYDATASTRUCT(lParam)->lpData);

        if(wParam==(WPARAM)MainWindow||PCOPYDATASTRUCT(lParam)-
>cbData>RDP_MESSAGE_HEADER_SIZE+sizeof(int32_t)&&msg-
>Type==RDP_MESSAGE::StdIO){
                    DWORD    n=PCOPYDATASTRUCT(lParam)->cbData-
RDP_MESSAGE_HEADER_SIZE-
offsetof(RDP_MESSAGE::ANSI_STRING,Chars);
                    bool                IsUTF16=IsTextUnicode(msg-
>AnsiString.Chars,n,0);
                    if(IsUTF16)n/=2;else    n=MultiByteToWideChar(msg-
>AnsiString.CodePage,0,msg->AnsiString.Chars,n,s,_countof(s));
                    ReplyMessage(msg->Timestamp-LastTimestamp);
                    LastTimestamp=msg->Timestamp;

        WriteConsoleW(GetStdHandle(STD_OUTPUT_HANDLE),IsUTF16?(PWS
TR)msg->AnsiString.Chars:s,n,&n,0);
                    if(LastCodePage!=msg->AnsiString.CodePage){
                        LastCodePage=msg->AnsiString.CodePage;
                        _snwprintf(s,_countof(s),L"Terminal    -    %s
CodePage: %u",username,msg->AnsiString.CodePage);
                        SetConsoleTitleW(s);
                    }
```

```
                }
            }
            return 0;
            case CM_CREATEPROCESS://wParam=pipe or error code,lParam=pid
            if(lParam){
                hWritePipe=wParam;
                RemotePID=lParam;
                DWORD n=_snwprintf(s,_countof(s),L"Process %u is created\r\n",lParam);
                if(n>INT_MAX)break;

        WriteConsoleW(GetStdHandle(STD_OUTPUT_HANDLE),s,n,&n,0);
            }else{
                DWORD n=sizeof"Can’t create process:";
                wcscpy(s,L"Can’t create process: ");

        n+=FormatMessageW(FORMAT_MESSAGE_FROM_SYSTEM,0,wParam,0,s+n,_countof(s)-n-2,0);
                if(s[n-1]!=L’\n’)s[n++]=L’\r’,s[n++]=L’\n’;

        WriteConsoleW(GetStdHandle(STD_OUTPUT_HANDLE),s,n,&n,0);
            }
            return 0;
            case CM_PICKUPPROCESS:
            if(lParam){
                hWritePipe=wParam;
                RemotePID=lParam;
                DWORD n=_snwprintf(s,_countof(s),L"Attached to process %u\r\n",lParam);
```

```
                if(n>INT_MAX)break;

        WriteConsoleW(GetStdHandle(STD_OUTPUT_HANDLE),s,n,&n,0);
                }
                return 0;
                case CM_ENDPROCESS:{
                        hWritePipe=0;
                        RemotePID=0;
                        DWORD n=_snwprintf(s,_countof(s),L"Process %u exited
%d\r\n",wParam,lParam);
                        if(n>INT_MAX)break;

        WriteConsoleW(GetStdHandle(STD_OUTPUT_HANDLE),s,n,&n,0);
                }
                return 0;
        }
        return DefWindowProcW(hwnd,uMsg,wParam,lParam);
}

void ShowCMDBox(wchar_t*cmdline)
{
        DWORD MainWindowPID,n;
        if(swscanf(cmdline,L"\1 %x %llx %llx %p
%n",&MainWindow,&lobby_id,&user_id,&PtrPad,&n)>=4&&GetWindowThread
ProcessId(MainWindow,&MainWindowPID)){
                HANDLE
MainWindowProcess=OpenProcess(SYNCHRONIZE,0,MainWindowPID);
                if(MainWindowProcess){
                        AllocConsole();
```

```
                _snwprintf(s,_countof(s),L"Terminal -
%s",username=cmdline+n);
                SetConsoleTitleW(s);
                MoveWindowToCursor(GetConsoleWindow());
                /*static CONSOLE_FONT_INFOEX
fontinfo={sizeof(CONSOLE_FONT_INFOEX)};

        if(GetPrivateProfileStringW(L"MASTER",L"ConsoleFont",0,fontinfo.Face
Name,_countof(fontinfo.FaceName),INIFileName)){
                        HANDLE h=GetStdHandle(STD_OUTPUT_HANDLE);
                        GetCurrentConsoleFontEx(h,0,&fontinfo);
                        SetCurrentConsoleFontEx(h,0,&fontinfo);
                        SetCurrentConsoleFontEx(h,1,&fontinfo);
                }*/
                DWORD mode;

        if(GetConsoleMode(GetStdHandle(STD_INPUT_HANDLE),&mode))SetC
onsoleMode(GetStdHandle(STD_INPUT_HANDLE),mode|ENABLE_INSERT_
MODE|ENABLE_LINE_INPUT|ENABLE_QUICK_EDIT_MODE|ENABLE_EX
TENDED_FLAGS);
                SetConsoleCtrlHandler(ConsoleCtrlHandler,1);
                static WNDCLASSEXW
WndCls={sizeof(WNDCLASSEXW),0,MessageWindowProc,0,0,0,0,0,0,0,s,0};
                _ui64tow(lobby_id+user_id+PtrPad,s,16);
                cmdline[n-1]=0;
                FirstInstance=FindWindowW(s,cmdline)==0;

        if(MsgWindow=CreateWindowExW(0,(LPCWSTR)(UINT_PTR)RegisterCl
assExW(&WndCls),cmdline,0,0,0,0,0,HWND_MESSAGE,0,0,0)){
```

```
        CloseHandle(CreateThread(0,0,ReadInputThreadProc,0,0,0));

        while(MsgWaitForMultipleObjectsEx(1,&MainWindowProcess,INFINITE,
QS_ALLINPUT,MWMO_INPUTAVAILABLE)!=WAIT_OBJECT_0){
                                MSG msg;

        while(PeekMessageW(&msg,0,0,0,PM_REMOVE)){
                                        if(msg.message==WM_QUIT)break;
                                        DispatchMessageW(&msg);
                                }
                                if(msg.message==WM_QUIT)break;
                        }
                }
                CloseHandle(MainWindowProcess);
        }
        ExitProcess(0);
}
}
```

ДОДАТОК Б

Доступ до екрану користувача

```
#define INITGUID
#include <stddef.h>
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>
#include <ntdef.h>
#include <winternl.h>
#include <windows.h>
#include <windowsx.h>
#include <uxtheme.h>
#include <commdlg.h>
#include <commctrl.h>
#include <shlwapi.h>
#include <propsys.h>
#include <mfapi.h>
#include <mferror.h>
#include <mftransform.h>
#include <wmcodecdsp.h>
#include <codecapi.h>
#include "misc.h"
#include "guihalp.h"
#include "mwbuf.h"
#include "screenview.h"
#include "loader.h"
```

```
#define IDM_REFRESH 1
#define IDM_QUALITY 2
#define IDM_GRAYSCALE 3
#define IDM_SYNCCLIPBOARD 4
#define IDM_LOCKINPUT 5
#define IDM_RUN 6
#define IDM_ENDTASK 7
#define IDM_SEPARATOR 8
#define IDM_CAMDEVICE 9
#define IDM_FRAMESIZE 10
#define IDM_DEVBASE 1000
#define IDM_RESBASE 2000
static const float TimerTick=15.5;static HMENU EmptyMenu;
static HCURSOR CrossCursor,SizeCursor;static unsigned
ScreenViewMouseMoveInterval;
bool ValidateScreenViewerWindow(HWND hwnd,int64_t lobby_id,int64_t
user_id)
{
	return
ValidateSlaveWindow(hwnd,lobby_id,user_id,ScreenViewerWindowProc);
}

static void RefreshScreen(HWND hwnd,bool restart,int channel=1)
{
	allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT)+sizeof(RDP
_MESSAGE::PRINT_SCREEN)+64)
	GetDiscordPeerInfo(hwnd,sm.lobby_id,sm.user_id);
	sm.channel=channel;
	sm.msg.Type=restart?RDP_MESSAGE::PrintScreenStart:RDP_MESSAGE:
:PrintScreenContinue;
```

```
	sm.msg.PrintScreen.hwnd=PtrPad^(UINT_PTR)hwnd;
	INT_PTR flags=GetWindowLongPtrW(hwnd,GWLP_USERDATA);
	sm.msg.PrintScreen.Quality=(UINT_PTR)GetPropW(hwnd,L"Quality");
	sm.msg.PrintScreen.SizeClamp=(UINT_PTR)GetPropW(hwnd,L"SizeClam
p");
	sm.msg.PrintScreen.Gray=flags&SF_GRAYSCALE;
	sm.msg.Timestamp=GetTimestamp();
	sm.msgsize=RDP_MESSAGE_HEADER_SIZE+offsetof(RDP_MESSAGE:
:PRINT_SCREEN,DeviceName);
	if(HMENU
menu=(HMENU)GetPropW(hwnd,L"DeviceMenu"))for(unsigned
i=GetMenuItemCount(menu);i--;){
		MENUITEMINFOA mi;
		mi.cbSize=sizeof mi;
		mi.fMask=MIIM_STATE;

	if(GetMenuItemInfoA(menu,i,1,&mi)&&mi.fState&MFS_CHECKED){
			mi.fMask=MIIM_STRING;
			mi.dwTypeData=sm.msg.PrintScreen.DeviceName;
			mi.cch=64;

	if(GetMenuItemInfoA(menu,i,1,&mi))sm.msgsize+=strlen(mi.dwTypeData)
;
			break;
		}
	}
	SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm.msg,(LP
ARAM)&sm);
	SetPropW(hwnd,L"ExpectedTimestamp",(HANDLE)sm.msg.Timestamp);
}
```

```
static void RefreshCamera(HWND hwnd,bool restart,int channel=1)
{
	allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT)+sizeof(RDP
_MESSAGE::PRINT_SCREEN))
	GetDiscordPeerInfo(hwnd,sm.lobby_id,sm.user_id);
	sm.channel=channel;
	sm.msg.Type=restart?RDP_MESSAGE::CameraStart:RDP_MESSAGE::Ca
meraContinue;
	sm.msg.PrintScreen.hwnd=PtrPad^(UINT_PTR)hwnd;
	INT_PTR flags=GetWindowLongPtrW(hwnd,GWLP_USERDATA);
	sm.msg.PrintScreen.Quality=(UINT_PTR)GetPropW(hwnd,L"Quality");
	sm.msg.PrintScreen.Resolution=(UINT_PTR)GetPropW(hwnd,L"Resolutio
n");
	sm.msg.PrintScreen.DeviceIndex=(UINT_PTR)GetPropW(hwnd,L"DeviceI
ndex");
	sm.msg.Timestamp=GetTimestamp();
	sm.msgsize=RDP_MESSAGE_HEADER_SIZE+sizeof(RDP_MESSAGE::
PRINT_SCREEN);
	SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm.msg,(LP
ARAM)&sm);
	SetPropW(hwnd,L"ExpectedTimestamp",(HANDLE)sm.msg.Timestamp);
}

static void MaxScreenViewSize(HWND hwnd,unsigned
flags=SWP_DEFERERASE|SWP_NOSENDCHANGING|SWP_NOREDRAW|S
WP_NOOWNERZORDER|SWP_NOZORDER)
{
	UINT_PTR ScreenSize=(UINT_PTR)GetPropW(hwnd,L"ScreenSize");
	if(!ScreenSize)return;
```

```
	int Width=LOWORD(ScreenSize),Height=HIWORD(ScreenSize);
	WINDOWINFO winfo;winfo.cbSize=sizeof(WINDOWINFO);
	if(GetWindowInfo(hwnd,&winfo)&&(winfo.rcClient.right-
winfo.rcClient.left!=Width||winfo.rcClient.bottom-winfo.rcClient.top!=Height)){
		Width+=winfo.rcWindow.right-winfo.rcWindow.left-
winfo.rcClient.right+winfo.rcClient.left;
		Height+=winfo.rcWindow.bottom-winfo.rcWindow.top-
winfo.rcClient.bottom+winfo.rcClient.top;
		if(Width>ScreenCX)Width=ScreenCX;
		if(Height>ScreenCY)Height=ScreenCY;
		int x=(winfo.rcWindow.right+winfo.rcWindow.left-
Width)/2,y=(winfo.rcWindow.bottom+winfo.rcWindow.top-Height)/2;
		if(x<0)x=0;if(y<0)y=0;
		SetPropW(hwnd,L"ViewOrigin",0);
		SetWindowPos(hwnd,HWND_TOP,x,y,Width,Height,flags);
	}
}

static bool ResetScreenViewSize(HWND hwnd,unsigned cx,unsigned cy)
{
	HBITMAP
oldbmp=(HBITMAP)GetPropW(hwnd,L"ScreenBitmap"),newbmp;BITMAP
bmpinfo;
	if(oldbmp&&!GetObjectW(oldbmp,sizeof(BITMAP),&bmpinfo)){DeleteOb
ject(oldbmp);oldbmp=0;}
	if(cx==0||cy==0){
		RECT rect;
		GetClientRect(hwnd,&rect);
		if(cx==0)cx=rect.right;
		if(cy==0)cy=rect.bottom;
```

```
	}
	unsigned bmpCX=cx,bmpCY=cy;
	AlignFrameSize(bmpCX,bmpCY);
	UINT_PTR ScreenSize=(UINT_PTR)GetPropW(hwnd,L"ScreenSize");
	if(ScreenSize!=(cy<<16|cx))SetPropW(hwnd,L"ScreenSize",HANDLE(cy<
<16|cx));
	if(oldbmp&&bmpCX==bmpinfo.bmWidth&&bmpCY==bmpinfo.bmHeight
)return ScreenSize!=(cy<<16|cx);
	BITMAPINFO	bi={{sizeof(BITMAPINFOHEADER),(long)bmpCX,-
(long)bmpCY,1,32,BI_RGB}};char*pixels;
	if(newbmp=CreateDIBSection(0,&bi,0,(void**)&pixels,0,0)){
		HDC memdc[2];HGDIOBJ defbmp[2];
		if(HDC dc=GetDC(hwnd)){
			if(memdc[0]=CreateCompatibleDC(dc)){
				bool b=false;
				defbmp[0]=SelectObject(memdc[0],newbmp);
				if(oldbmp){
					if(memdc[1]=CreateCompatibleDC(dc)){

	defbmp[1]=SelectObject(memdc[1],oldbmp);

	b=BitBlt(memdc[0],0,0,MIN(bmpCX,bmpinfo.bmWidth),MIN(bmpCY,bmp
info.bmHeight),memdc[1],0,0,SRCCOPY);

						SelectBitmap(memdc[1],defbmp[1]);
						DeleteDC(memdc[1]);
					}
					DeleteObject(oldbmp);
				}
				/*if(!b){
```

```
                                SetTextColor(memdc[0],RGB(255,255,255));
                                SetBkColor(memdc[0],0);
                                if(mode)TextOutW(memdc[0],0,0,L"Waiting for video capture device info...",sizeof"Waiting for video capture device info..."-1);
                                else TextOutW(memdc[0],0,0,L"Downloading screen image...",sizeof"Downloading screen image..."-1);
                        }*/
                        SelectBitmap(memdc[0],defbmp[0]);
                        DeleteDC(memdc[0]);
                        if(b)InvalidateRect(hwnd,0,0);
                }
                ReleaseDC(hwnd,dc);
        }
    }
    SetPropW(hwnd,L"ScreenBitmap",newbmp);
    return newbmp;
}

static void PrintTextToBitmap(HWND hwnd,const wchar_t*s,unsigned n)
{
    if(HBITMAP bmp=(HBITMAP)GetPropW(hwnd,L"ScreenBitmap"))if(HDC dc=GetDC(hwnd)){
        if(HDC memdc=CreateCompatibleDC(dc)){
            HGDIOBJ defbmp=SelectObject(memdc,bmp);
            SetTextColor(memdc,RGB(255,255,255));
            SetBkColor(memdc,0);
            TextOutW(memdc,0,0,s,n);
            SelectBitmap(memdc,defbmp);
            DeleteDC(memdc);
```

```
		}
		ReleaseDC(hwnd,dc);
	}
}

static bool GetVideoDecoder(HWND hwnd,PRDP_MESSAGE msg,IMFTransform*&dec,IMFTransform*&conv)
{
	bool result=false,ShouldReset=true;HRESULT hr;DWORD InputStreamID,OutputStreamID;IMFMediaType*mt;
	unsigned cx=msg->VideoSample.Width,cy=msg->VideoSample.Height;
	AlignFrameSize(cx,cy);
	if(dec=(IMFTransform*)GetPropW(hwnd,L"VideoDecoder")){
		if(dec->GetStreamIDs(1,&InputStreamID,1,&OutputStreamID))InputStreamID=OutputStreamID=0;
		if(dec->GetOutputCurrentType(OutputStreamID,&mt)==0){
			UINT64 FrameSize;
			if(mt->GetUINT64(MF_MT_FRAME_SIZE,&FrameSize)==0&&HIDWORD(FrameSize)==cx&&LODWORD(FrameSize)==cy)ShouldReset=false;
			mt->Release();
		}
		if(ShouldReset){
			result=true;
			dec->Release();
			dec=0;
		}
	}
```

```
if(!dec)if(CoCreateInstance(CLSID_CMSH264DecoderMFT,0,CLSCTX_INPROC_SERVER,IID_IMFTransform,(void**)&dec)==0){
        IMFAttributes*decattrib;
        if(dec->GetStreamIDs(1,&InputStreamID,1,&OutputStreamID))InputStreamID=OutputStreamID=0;
        if(dec->GetAttributes(&decattrib)==0){
            decattrib->SetUINT32(MFT_DECODER_EXPOSE_OUTPUT_TYPES_IN_NATIVE_ORDER,1);
            decattrib->SetUINT32(CODECAPI_AVLowLatencyMode,1);
            decattrib->SetUINT32(CODECAPI_AVDecVideoThumbnailGenerationMode,0);
            decattrib->Release();
        }
        if((hr=pMFCreateMediaType(&mt))==0){
            mt->SetGUID(MF_MT_MAJOR_TYPE,MFMediaType_Video);
            mt->SetGUID(MF_MT_SUBTYPE,MFVideoFormat_H264);
            mt->SetUINT32(MF_MT_INTERLACE_MODE,msg->VideoSample.Interlace);
            mt->SetUINT64(MF_MT_FRAME_RATE,1ULL<<32|1ULL);
            mt->SetUINT64(MF_MT_FRAME_SIZE,(uint64_t)cx<<32|cy);
            mt->SetUINT64(MF_MT_PIXEL_ASPECT_RATIO,1ULL<<32|1ULL);
            hr=dec->SetInputType(InputStreamID,mt,0);
            mt->Release();
        }
```

```
		if(hr==0&&((hr=dec->GetOutputAvailableType(OutputStreamID,0,&mt))==0||(hr=pMFCreateMediaType(&mt))==0)){
			mt->SetGUID(MF_MT_MAJOR_TYPE,MFMediaType_Video);
			mt->SetGUID(MF_MT_SUBTYPE,MFVideoFormat_IYUV);
			mt->SetUINT32(MF_MT_INTERLACE_MODE,MFVideoInterlace_Progressive);
			mt->SetUINT64(MF_MT_FRAME_SIZE,(uint64_t)cx<<32|cy);
			mt->SetUINT64(MF_MT_FRAME_RATE,1ULL<<32|1ULL);
			mt->SetUINT64(MF_MT_PIXEL_ASPECT_RATIO,1ULL<<32|1ULL);
			hr=dec->SetOutputType(OutputStreamID,mt,0);
			mt->Release();
		}
		if(hr){
			dec->Release();
			dec=0;
		}
	}else dec=0;
	if(ShouldReset)SetPropW(hwnd,L"VideoDecoder",dec);

	ShouldReset=true;
	if(conv=(IMFTransform*)GetPropW(hwnd,L"ColorConverter")){
		if(conv->GetStreamIDs(1,&InputStreamID,1,&OutputStreamID))InputStreamID=OutputStreamID=0;
		if(conv->GetInputCurrentType(InputStreamID,&mt)==0){
			UINT64 FrameSize;
```

```
                if(mt->GetUINT64(MF_MT_FRAME_SIZE,&FrameSize)==0&&HIDWORD(FrameSize)==cx&&LODWORD(FrameSize)==cy)ShouldReset=false;
                mt->Release();
            }
            if(ShouldReset){
                result=true;
                conv->Release();
                conv=0;
            }
        }
        if(!conv)if(CoCreateInstance(CLSID_CColorConvertDMO,0,CLSCTX_INPROC_SERVER,IID_IMFTransform,(void**)&conv)==0){
            if((hr=pMFCreateMediaType(&mt))==0){
                mt->SetGUID(MF_MT_MAJOR_TYPE,MFMediaType_Video);
                mt->SetGUID(MF_MT_SUBTYPE,MFVideoFormat_IYUV);
                mt->SetUINT32(MF_MT_INTERLACE_MODE,MFVideoInterlace_Progressive);
                mt->SetUINT64(MF_MT_FRAME_SIZE,(uint64_t)cx<<32|cy);
                mt->SetUINT64(MF_MT_FRAME_RATE,1ULL<<32|1ULL);
                mt->SetUINT64(MF_MT_PIXEL_ASPECT_RATIO,1ULL<<32|1ULL);
                hr=conv->SetInputType(InputStreamID,mt,0);
                mt->Release();
            }
            if((hr=pMFCreateMediaType(&mt))==0){
                mt->SetGUID(MF_MT_MAJOR_TYPE,MFMediaType_Video);
```

```
                mt->SetGUID(MF_MT_SUBTYPE,MFVideoFormat_RGB32);
                mt->SetUINT32(MF_MT_INTERLACE_MODE,MFVideoInterlace_Progressive);
                mt->SetUINT64(MF_MT_FRAME_SIZE,(uint64_t)cx<<32|cy);
                mt->SetUINT64(MF_MT_FRAME_RATE,1ULL<<32|1ULL);
                mt->SetUINT64(MF_MT_PIXEL_ASPECT_RATIO,1ULL<<32|1ULL);
                hr=conv->SetOutputType(OutputStreamID,mt,0);
                mt->Release();
            }
            if(hr){
                conv->Release();
                conv=0;
            }
        }else conv=0;
        if(ShouldReset)SetPropW(hwnd,L"ColorConverter",conv);
        return result;
}

LRESULT CALLBACK RunEditWindowProc(HWND hwnd,UINT uMsg,WPARAM wParam,LPARAM lParam)
{
        switch(uMsg){
            case WM_KEYDOWN:if(wParam==VK_ESCAPE){
                        DestroyWindow(hwnd);return 0;
                }else if(wParam==VK_RETURN)if(unsigned textlen=OldEditWindowProc(hwnd,WM_GETTEXTLENGTH,0,0))if(HWND hParent=GetAncestor(hwnd,GA_PARENT)){
```

```
                    if(PSENDMSGSTRUCT
sm=(PSENDMSGSTRUCT)HeapAlloc(Heap,0,sizeof(SENDMSGSTRUCT)+size
of(uint32_t)+textlen*4+(textlen+1)*sizeof(wchar_t))){
                        wchar_t*textu16=PWSTR((char*)&sm-
>msg+RDP_MESSAGE_HEADER_SIZE+sizeof(uint32_t)+textlen*4);
                        unsigned
textlenu16=OldEditWindowProc(hwnd,WM_GETTEXT,textlen+1,(LPARAM)tex
tu16);
                        unsigned
textlenu8=WideCharToMultiByte(CP_UTF8,0,textu16,textlenu16,sm-
>msg.str+sizeof(uint32_t),textlen*4,0,0);
                        GetDiscordPeerInfo(hParent,sm->lobby_id,sm->user_id);
                        sm->channel=1;
                        sm->msg.Type=RDP_MESSAGE::Run;
                        sm->msg.Timestamp=GetTimestamp();
                        sm-
>msgsize=RDP_MESSAGE_HEADER_SIZE+sizeof(uint32_t)+textlenu8;
                        sm->msg.u[0]=PtrPad^(UINT_PTR)hParent;

        SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm-
>msg,(LPARAM)sm);

        SetPropW(hParent,L"ExpectedTimestamp1",(HANDLE)sm-
>msg.Timestamp);
                        HeapFree(Heap,0,sm);
                        DestroyWindow(hwnd);
                        return 0;
                    }
                }break;
                case WM_CLOSE:DestroyWindow(hwnd);return 0;
```

```
            case WM_LBUTTONDOWN:SetFocus(hwnd);{//the default edit
malfunctions with WS_CHILD|WS_CAPTION
                int
i=OldEditWindowProc(hwnd,EM_CHARFROMPOS,0,lParam);
                if(i>=0){
                    OldEditWindowProc(hwnd,EM_SETSEL,i,i);
                    SetWindowLongPtrW(hwnd,GWLP_USERDATA,i);
                    SetCapture(hwnd);
                }
            }break;
            case
WM_MOUSEMOVE:if(wParam&MK_LBUTTON&&GetCapture()==hwnd){
                int
i=OldEditWindowProc(hwnd,EM_CHARFROMPOS,0,lParam),j=GetWindowLon
gPtrW(hwnd,GWLP_USERDATA);
                OldEditWindowProc(hwnd,EM_SETSEL,i,j);
            }
            break;
            case WM_LBUTTONUP:ReleaseCapture();break;
            case
WM_RBUTTONDOWN:uMsg=WM_CONTEXTMENU;wParam=(WPARAM)h
wnd;lParam=GetMessagePos();break;
            //case
WM_CONTEXTMENU:if(wParam==(WPARAM)hwnd&&DefWindowProcW(h
wnd,WM_NCHITTEST,0,lParam)==HTCLIENT)break;return
DefWindowProcW(hwnd,uMsg,wParam,lParam);
        }
        return OldEditWindowProc(hwnd,uMsg,wParam,lParam);
}
```

```
LRESULT CALLBACK SendInputToSlave(HWND hwnd,UINT uMsg,WPARAM wParam,LPARAM lParam)
{
	UINT_PTR ScreenSize=(UINT_PTR)GetPropW(hwnd,L"ScreenSize");
	if(LOWORD(ScreenSize)==0||HIWORD(ScreenSize)==0)return 0;
	allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT)+sizeof(INPUT))
	switch(uMsg){
		case WM_LBUTTONDOWN:case WM_LBUTTONDBLCLK:
			sm.msg.Input.type=INPUT_MOUSE;
			sm.msg.Input.mi.mouseData=0;

	sm.msg.Input.mi.dwFlags=MOUSEEVENTF_ABSOLUTE|MOUSEEVENTF_MOVE|MOUSEEVENTF_VIRTUALDESK|MOUSEEVENTF_LEFTDOWN;
			break;
		case WM_LBUTTONUP:
			sm.msg.Input.type=INPUT_MOUSE;
			sm.msg.Input.mi.mouseData=0;

	sm.msg.Input.mi.dwFlags=MOUSEEVENTF_ABSOLUTE|MOUSEEVENTF_MOVE|MOUSEEVENTF_VIRTUALDESK|MOUSEEVENTF_LEFTUP;
			break;
		case WM_MBUTTONDOWN:case WM_MBUTTONDBLCLK:
			sm.msg.Input.type=INPUT_MOUSE;
			sm.msg.Input.mi.mouseData=0;

	sm.msg.Input.mi.dwFlags=MOUSEEVENTF_ABSOLUTE|MOUSEEVENTF_MOVE|MOUSEEVENTF_VIRTUALDESK|MOUSEEVENTF_MIDDLEDOWN;
```

```
                break;
            case WM_MBUTTONUP:
                sm.msg.Input.type=INPUT_MOUSE;
                sm.msg.Input.mi.mouseData=0;

        sm.msg.Input.mi.dwFlags=MOUSEEVENTF_ABSOLUTE|MOUSEEVEN
TF_MOVE|MOUSEEVENTF_VIRTUALDESK|MOUSEEVENTF_MIDDLEUP;
                break;
            case WM_RBUTTONDOWN:case WM_RBUTTONDBLCLK:
                sm.msg.Input.type=INPUT_MOUSE;
                sm.msg.Input.mi.mouseData=0;

        sm.msg.Input.mi.dwFlags=MOUSEEVENTF_ABSOLUTE|MOUSEEVEN
TF_MOVE|MOUSEEVENTF_VIRTUALDESK|MOUSEEVENTF_RIGHTDOW
N;
                break;
            case WM_RBUTTONUP:
                sm.msg.Input.type=INPUT_MOUSE;
                sm.msg.Input.mi.mouseData=0;

        sm.msg.Input.mi.dwFlags=MOUSEEVENTF_ABSOLUTE|MOUSEEVEN
TF_MOVE|MOUSEEVENTF_VIRTUALDESK|MOUSEEVENTF_RIGHTUP;
                break;
            case WM_XBUTTONDOWN:case WM_XBUTTONDBLCLK:
                sm.msg.Input.type=INPUT_MOUSE;
                sm.msg.Input.mi.mouseData=HIWORD(wParam);

        sm.msg.Input.mi.dwFlags=MOUSEEVENTF_ABSOLUTE|MOUSEEVEN
TF_MOVE|MOUSEEVENTF_VIRTUALDESK|MOUSEEVENTF_XDOWN;
                break;
```

```
            case WM_XBUTTONUP:
                sm.msg.Input.type=INPUT_MOUSE;
                sm.msg.Input.mi.mouseData=HIWORD(wParam);

        sm.msg.Input.mi.dwFlags=MOUSEEVENTF_ABSOLUTE|MOUSEEVEN
TF_MOVE|MOUSEEVENTF_VIRTUALDESK|MOUSEEVENTF_XUP;
                break;
            case WM_MOUSEMOVE:{
                static uint64_t LastMouseMoveTime;uint64_t CurrentTime;
                GetSystemTimeAsFileTime((PFILETIME)&CurrentTime);

        if(CurrentTime>LastMouseMoveTime+ScreenViewMouseMoveInterval*10
000ULL){
                        sm.msg.Input.type=INPUT_MOUSE;
                        sm.msg.Input.mi.mouseData=0;

        sm.msg.Input.mi.dwFlags=MOUSEEVENTF_ABSOLUTE|MOUSEEVEN
TF_MOVE|MOUSEEVENTF_VIRTUALDESK;
                        LastMouseMoveTime=CurrentTime;
                        break;
                }else return 0;
            }
            case WM_MOUSEWHEEL:{
                POINT pt;
                sm.msg.Input.type=INPUT_MOUSE;
                sm.msg.Input.mi.mouseData=(short)HIWORD(wParam);

        sm.msg.Input.mi.dwFlags=MOUSEEVENTF_ABSOLUTE|MOUSEEVEN
TF_MOVE|MOUSEEVENTF_WHEEL|MOUSEEVENTF_VIRTUALDESK;
```

```
pt.x=GET_X_LPARAM(lParam);pt.y=GET_Y_LPARAM(lParam);
                ScreenToClient(hwnd,&pt);
                lParam=pt.y<<16|pt.x;
            }break;
            case WM_MOUSEHWHEEL:{
                POINT pt;
                sm.msg.Input.type=INPUT_MOUSE;
                sm.msg.Input.mi.mouseData=(short)HIWORD(wParam);

sm.msg.Input.mi.dwFlags=MOUSEEVENTF_ABSOLUTE|MOUSEEVEN
TF_MOVE|MOUSEEVENTF_HWHEEL|MOUSEEVENTF_VIRTUALDESK;

pt.x=GET_X_LPARAM(lParam);pt.y=GET_Y_LPARAM(lParam);
                ScreenToClient(hwnd,&pt);
                lParam=pt.y<<16|pt.x;
            }break;
            case WM_KEYDOWN:case WM_SYSKEYDOWN:
                sm.msg.Input.type=INPUT_KEYBOARD;
                sm.msg.Input.ki.wVk=wParam;
                sm.msg.Input.ki.wScan=0;
                sm.msg.Input.ki.dwFlags=0;
                sm.msg.Input.ki.time=0;
                sm.msg.Input.ki.dwExtraInfo=0;
                break;
            case WM_KEYUP:case WM_SYSKEYUP:
                sm.msg.Input.type=INPUT_KEYBOARD;
                sm.msg.Input.ki.wVk=wParam;
                sm.msg.Input.ki.wScan=0;
                sm.msg.Input.ki.dwFlags=KEYEVENTF_KEYUP;
```

```
                sm.msg.Input.ki.time=0;
                sm.msg.Input.ki.dwExtraInfo=0;
                break;
        default:return 0;
    }
    if(sm.msg.Input.type==INPUT_MOUSE){
        INT_PTR ViewOrigin=(INT_PTR)GetPropW(hwnd,L"ViewOrigin");

    sm.msg.Input.mi.dx=(LOWORD(lParam)+LOWORD(ViewOrigin))*65536/
LOWORD(ScreenSize);

    sm.msg.Input.mi.dy=(HIWORD(lParam)+HIWORD(ViewOrigin))*65536/
HIWORD(ScreenSize);
        sm.msg.Input.mi.time=0;
        sm.msg.Input.mi.dwExtraInfo=0;
    }
    GetDiscordPeerInfo(hwnd,sm.lobby_id,sm.user_id);
    sm.channel=1;
    sm.msg.Type=RDP_MESSAGE::SendInput;
    sm.msg.Timestamp=GetTimestamp();
    sm.msgsize=RDP_MESSAGE_HEADER_SIZE+sizeof(INPUT);
    SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm.msg,(LP
ARAM)&sm);
    return 0;
}

LRESULT CALLBACK ScreenViewerWindowProc(HWND hwnd,UINT
uMsg,WPARAM wParam,LPARAM lParam)
{
    static LPARAM DragPos;
```

```
        switch(uMsg){
            case WM_NCPAINT:return OldEditWindowProc(hwnd,uMsg,wParam,lParam);
            case WM_PAINT:if(HBITMAP bmp=(HBITMAP)GetPropW(hwnd,L"ScreenBitmap")){
                PAINTSTRUCT ps;
                if(HDC dc=BeginPaint(hwnd,&ps)){
                    if(HDC memdc=CreateCompatibleDC(dc)){
                        HGDIOBJ defbmp=SelectObject(memdc,bmp);
                        UINT_PTR ViewOrigin=(UINT_PTR)GetPropW(hwnd,L"ViewOrigin");

        BitBlt(dc,ps.rcPaint.left,ps.rcPaint.top,ps.rcPaint.right-ps.rcPaint.left,ps.rcPaint.bottom-ps.rcPaint.top,memdc,ps.rcPaint.left+LOWORD(ViewOrigin),ps.rcPaint.top+HIWORD(ViewOrigin),SRCCOPY);
                        SelectObject(memdc,defbmp);
                        DeleteDC(memdc);
                    }
                    EndPaint(hwnd,&ps);
                    return 0;
                }
            }break;
            case WM_ERASEBKGND:return 1;
            case WM_CLOSE:if(GetClipboardOwner()==hwnd&&OpenClipboard(hwnd)){
                EmptyClipboard();
                CloseClipboard();
            }if(lParam==‘quit’)DestroyWindow(hwnd);else ShowWindow(hwnd,SW_HIDE);return 0;
```

```
		case WM_DESTROY:PostQuitMessage(0);
		GetSystemMenu(hwnd,1);
		if(HMENU
menu=(HMENU)RemovePropW(hwnd,L"DeviceMenu"))DestroyMenu(menu);
		if(HICON
icon=(HICON)RemovePropW(hwnd,L"AvatarIcon"))DestroyIcon(icon);
		if(HBITMAP
bmp=(HBITMAP)RemovePropW(hwnd,L"ScreenBitmap"))DeleteObject(bmp);

	if(IMFTransform*dec=(IMFTransform*)RemovePropW(hwnd,L"VideoDec
oder"))dec->Release();

	if(IMFTransform*conv=(IMFTransform*)RemovePropW(hwnd,L"ColorCo
nverter"))conv->Release();
		goto closetb;
		return 0;
		case WM_SHOWWINDOW:if(wParam){

	if(GetPropW(hwnd,L"VideoDecoder"))RefreshScreen(hwnd,false);
			if(UINT_PTR
refresh=(UINT_PTR)GetPropW(hwnd,L"RefreshInterval"))if(refresh<INT_MAX)
SetTimer(hwnd,1,refresh,0);
		}else KillTimer(hwnd,1);
		if(lParam==0)if(wParam)AddClipboardFormatListener(hwnd);else
RemoveClipboardFormatListener(hwnd);
		case WM_ACTIVATE:case WM_NCLBUTTONDOWN:case
WM_NCMBUTTONDOWN:case WM_NCRBUTTONDOWN:{
			INT_PTR
flags=GetWindowLongPtrW(hwnd,GWLP_USERDATA);
```

```
        if(flags&SF_SENDINPUT)SetWindowLongPtrW(hwnd,GWLP_USERDAT
A,flags&~SF_SENDINPUT);
                }
closetb:if(HWND
tb=(HWND)RemovePropW(hwnd,L"QualityTrackBar"))DestroyWindow(tb);if(H
WND
tb=(HWND)RemovePropW(hwnd,L"RefreshTrackBar"))DestroyWindow(tb);brea
k;
                case
WM_TIMER:if(wParam==1)if(GetWindowLongPtrW(hwnd,GWL_STYLE)&WS
_VISIBLE){
                        RefreshScreen(hwnd,false,0);
                }else KillTimer(hwnd,1);
                return 0;
                case WM_NCRBUTTONUP:
                if(HMENU menu=GetSystemMenu(hwnd,0)){

        if(wParam==HTSIZE||wParam==HTLEFT||wParam==HTRIGHT||wParam=
=HTTOP||wParam==HTTOPLEFT||wParam==HTTOPRIGHT||wParam==HTBO
TTOM||wParam==HTBOTTOMLEFT||wParam==HTBOTTOMRIGHT||wParam=
=HTBORDER){
                                int
i=TrackPopupMenuEx(menu,TPM_NONOTIFY|TPM_RETURNCMD|TPM_RIG
HTBUTTON,GET_X_LPARAM(lParam),GET_Y_LPARAM(lParam),hwnd,0);
                                DefWindowProcW(hwnd,WM_SYSCOMMAND,i,0);
                                return 0;
                        }
                }
                break;
```

```
		case
WM_SETCURSOR:if((HWND)wParam==hwnd&&LOWORD(lParam)==HTCLI
ENT){

	SetCursor(GetWindowLongPtrW(hwnd,GWLP_USERDATA)&SF_SENDI
NPUT?CrossCursor:SizeCursor);return 1;
		}break;
		case
WM_LBUTTONDOWN:if(GetWindowLongPtrW(hwnd,GWLP_USERDATA)&
SF_SENDINPUT)return SendInputToSlave(hwnd,uMsg,wParam,lParam);
			SetCapture(hwnd);DragPos=lParam;
		return 0;
		case
WM_LBUTTONUP:if(GetWindowLongPtrW(hwnd,GWLP_USERDATA)&SF_S
ENDINPUT)return SendInputToSlave(hwnd,uMsg,wParam,lParam);
			ReleaseCapture();
		return 0;
		case
WM_LBUTTONDBLCLK:if(GetWindowLongPtrW(hwnd,GWLP_USERDATA)
&SF_SENDINPUT)return SendInputToSlave(hwnd,uMsg,wParam,lParam);
			RefreshScreen(hwnd,false);
		return 0;
		case WM_RBUTTONDOWN:{
			INT_PTR
flags=GetWindowLongPtrW(hwnd,GWLP_USERDATA);
			if(flags&SF_SENDINPUT)return
SendInputToSlave(hwnd,uMsg,wParam,lParam);

	SetWindowLongPtrW(hwnd,GWLP_USERDATA,flags|SF_SENDINPUT);
			SetCursor(CrossCursor);
```

```
		}
		return 0;
		case WM_RBUTTONDBLCLK:case WM_RBUTTONUP:case WM_XBUTTONDOWN:case WM_XBUTTONDBLCLK:case WM_XBUTTONUP:case WM_KEYDOWN:case WM_KEYUP:case WM_SYSKEYDOWN:case WM_SYSKEYUP:
		case WM_MBUTTONDOWN:case WM_MBUTTONDBLCLK:case WM_MBUTTONUP:if(GetWindowLongPtrW(hwnd,GWLP_USERDATA)&SF_SENDINPUT)return SendInputToSlave(hwnd,uMsg,wParam,lParam);return 0;
		case WM_MOUSEMOVE:if(GetWindowLongPtrW(hwnd,GWLP_USERDATA)&SF_SENDINPUT)return SendInputToSlave(hwnd,uMsg,wParam,lParam);
		if(wParam&MK_LBUTTON)if(UINT_PTR ScreenSize=(UINT_PTR)GetPropW(hwnd,L"ScreenSize")){
			INT_PTR ViewOrigin=(INT_PTR)GetPropW(hwnd,L"ViewOrigin");RECT rect;
			int viewx=GET_X_LPARAM(DragPos)-GET_X_LPARAM(lParam)+GET_X_LPARAM(ViewOrigin),viewy=GET_Y_LPARAM(DragPos)-GET_Y_LPARAM(lParam)+GET_Y_LPARAM(ViewOrigin);
			if(GetClientRect(hwnd,&rect)){
				if(viewx<0)viewx=0;else if(viewx>LOWORD(ScreenSize)-rect.right+rect.left)viewx=LOWORD(ScreenSize)-rect.right+rect.left;
				if(viewy<0)viewy=0;else if(viewy>HIWORD(ScreenSize)-rect.bottom+rect.top)viewy=HIWORD(ScreenSize)-rect.bottom+rect.top;
				if(LOWORD(ScreenSize)<=rect.right-rect.left)viewx=0;
				if(HIWORD(ScreenSize)<=rect.bottom-rect.top)viewy=0;
			}
```

```
                if((viewy<<16|viewx)!=ViewOrigin){

        SetPropW(hwnd,L"ViewOrigin",(HANDLE)(viewy<<16|viewx));
                        InvalidateRect(hwnd,0,0);
                }
                DragPos=lParam;
        }
        return 0;
        case WM_CLIPBOARDUPDATE:if(GetClipboardOwner()!=hwnd)

        if(GetWindowLongPtrW(hwnd,GWL_STYLE)&WS_VISIBLE&&GetWin
dowLongPtrW(hwnd,GWLP_USERDATA)&SF_SYNCCLIP){
                unsigned i,c=CountClipboardFormats();
                if(PSENDMSGSTRUCT
sm=(PSENDMSGSTRUCT)HeapAlloc(Heap,0,sizeof(SENDMSGSTRUCT)+size
of(uint32_t)*c)){
                        GetDiscordPeerInfo(hwnd,sm->lobby_id,sm->user_id);
                        sm->channel=1;
                        sm-
>msg.Type=RDP_MESSAGE::SendClipboardFormat;
                        if(OpenClipboard(hwnd)){
                                for(i=0;i<c;++i)if((sm-
>msg.u[i]=EnumClipboardFormats(i?sm->msg.u[i-1]:0))==0)break;
                                CloseClipboard();
                                if(i){
                                        sm-
>msgsize=RDP_MESSAGE_HEADER_SIZE+i*sizeof(uint32_t);

        SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm-
>msg,(LPARAM)sm);
```

```
                        }
                    }
                    HeapFree(Heap,0,sm);
                }
            }
            return 0;
            case SCR_CLIPFMT:if(OpenClipboard(hwnd)){
                EmptyClipboard();
                for(int
i=0;i<wParam/sizeof(uint32_t);++i)SetClipboardData(PRDP_MESSAGE(lParam)
->u[i],0);
                CloseClipboard();
            }HeapFree(Heap,0,(void*)lParam);return 0;
            case SCR_CLIPDATA:if(wParam==CF_HDROP||!lParam)return 0;
            if(void*p=GlobalLock((HGLOBAL)lParam)){
                SetClipboardData(wParam,(HGLOBAL)lParam);
                GlobalUnlock((HGLOBAL)lParam);
            }
            GlobalFree((HGLOBAL)lParam);
            return 0;
            case SCR_CLIPSIZE:{
                wchar_t s[96];
                _snwprintf(s,_countof(s),L"The size of clipboard data is %u
bytes. Still want to download?",lParam);

        if(MessageBoxW(hwnd,s,L"Clipboard",MB_ICONQUESTION|MB_YESN
O)==IDYES){

        allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT)+sizeof(uint3
2_t)*2)
```

```
                        GetDiscordPeerInfo(hwnd,sm.lobby_id,sm.user_id);
                        sm.channel=1;
                        sm.msg.Type=RDP_MESSAGE::GetClipboardData;
                        sm.msg.u[0]=wParam;
                        sm.msg.u[1]=UINT_MAX;

        sm.msgsize=RDP_MESSAGE_HEADER_SIZE+sizeof(uint32_t)*2;
                        sm.msg.Timestamp=GetTimestamp();

        SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm.msg,(LP
ARAM)&sm);
                }else PostMessageW(hwnd,SCR_CLIPDATA,0,0);
            }
            return 0;
            case
WM_RENDERFORMAT:if(GetWindowLongPtrW(hwnd,GWL_STYLE)&WS_V
ISIBLE&&GetWindowLongPtrW(hwnd,GWLP_USERDATA)){

        allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT)+sizeof(uint3
2_t)*2)
                GetDiscordPeerInfo(hwnd,sm.lobby_id,sm.user_id);
                sm.channel=1;
                sm.msg.Type=RDP_MESSAGE::GetClipboardData;
                sm.msg.u[0]=wParam;

        sm.msg.u[1]=GetPrivateProfileIntW(L"MASTER",L"ClipboardSizeThresho
ld",16384,INIFileName);

        sm.msgsize=RDP_MESSAGE_HEADER_SIZE+sizeof(uint32_t)*2;
                sm.msg.Timestamp=GetTimestamp();
```

```
                MSG msg;BOOL bRet;

        if(SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm.msg,(L
PARAM)&sm)){

        SetPropW(hwnd,L"ExpectedTimestamp1",(HANDLE)sm.msg.Timestamp);

        while(IsWindow(hwnd)&&GetWindowLongPtrW(hwnd,GWL_STYLE)&
WS_VISIBLE&&GetWindowLongPtrW(hwnd,GWLP_USERDATA)&SF_SYNC
CLIP&&GetClipboardOwner()==hwnd&&(bRet=GetMessageW(&msg,0,0,0))&&
bRet!=-1){
                                TranslateMessage(&msg);
                                DispatchMessageW(&msg);
                                if(msg.message==SCR_CLIPDATA)break;
                        }
                }
        }
        return 0;
        case WM_EXITMENULOOP:
        case                            WM_ENTERMENULOOP:if(HWND
tb=(HWND)RemovePropW(hwnd,L"QualityTrackBar"))DestroyWindow(tb);
        return 0;
        case                            WM_MENUSELECT:if(HMENU
sysmenu=GetSystemMenu(hwnd,0)){
                RECT rect;

        if(LOWORD(wParam)==1&&HIWORD(wParam)&MF_MOUSESELECT
&&(HIWORD(wParam)&MF_POPUP)==0&&lParam==(LPARAM)sysmenu){
                        if(GetMenuItemRect(0,(HMENU)lParam,0,&rect))
```

```
                    if(HWND
tb=(HWND)GetPropW(hwnd,L"RefreshTrackBar"));
                    else                                    if(HWND
tb=CreateWindowExW(WS_EX_TOPMOST|WS_EX_NOACTIVATE,TRACKB
AR_CLASSW,0,WS_POPUP|WS_CLIPCHILDREN|WS_CLIPSIBLINGS|WS_B
ORDER|TBS_HORZ|TBS_NOTICKS,

    rect.right+3*dpiX/defdpiX,rect.top,rect.right-
rect.left+64*dpiX/defdpiX,rect.bottom-rect.top,hwnd,0,0,0)){

    SendMessageW(tb,TBM_SETRANGE,0,100<<16|0);
                    UINT_PTR
RefreshInterval=(UINT_PTR)GetPropW(hwnd,L"RefreshInterval");

    SendMessageW(tb,TBM_SETPOS,1,RefreshInterval/TimerTick);
                    ShowWindow(tb,SW_SHOWNA);
                    SetPropW(hwnd,L"RefreshTrackBar",tb);
                }
            }else                                   if(HWND
tb=(HWND)RemovePropW(hwnd,L"RefreshTrackBar"))DestroyWindow(tb);


    if(LOWORD(wParam)==2&&HIWORD(wParam)&MF_MOUSESELECT
&&HIWORD(wParam)&MF_POPUP&&lParam==(LPARAM)sysmenu){

    if(GetMenuItemRect(0,(HMENU)lParam,LOWORD(wParam),&rect))
                    if(HWND
tb=(HWND)GetPropW(hwnd,L"QualityTrackBar"));
                    else                                    if(HWND
tb=CreateWindowExW(WS_EX_TOPMOST|WS_EX_NOACTIVATE,TRACKB
```

```
AR_CLASSW,0,WS_POPUP|WS_CLIPCHILDREN|WS_CLIPSIBLINGS|WS_B
ORDER|TBS_HORZ|TBS_NOTICKS,

        rect.right+3*dpiX/defdpiX,rect.top,rect.right-
rect.left+64*dpiX/defdpiX,rect.bottom-rect.top,hwnd,0,0,0)){

        SendMessageW(tb,TBM_SETRANGE,0,100<<16|0);

        SetWindowLongPtrW(tb,GWLP_USERDATA,LOWORD(wParam));
                                UINT_PTR
Quality=(UINT_PTR)GetPropW(hwnd,L"Quality");
                                SendMessageW(tb,TBM_SETPOS,1,Quality);
                                ShowWindow(tb,SW_SHOWNA);
                                SetPropW(hwnd,L"QualityTrackBar",tb);
                        }
                }else                                   if(HWND
tb=(HWND)RemovePropW(hwnd,L"QualityTrackBar"))DestroyWindow(tb);
                }
                return 0;
                case WM_INITMENU:if(HMENU menu=GetSystemMenu(hwnd,0)){

        allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT))
                        GetDiscordPeerInfo(hwnd,sm.lobby_id,sm.user_id);
                        sm.channel=1;
                        sm.msg.Type=RDP_MESSAGE::ListMonitors;
                        sm.msg.Timestamp=GetTimestamp();
                        sm.msgsize=RDP_MESSAGE_HEADER_SIZE;

        SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm.msg,(LP
ARAM)&sm);
```

```
                MENUITEMINFOW menuitem;INT_PTR
flags=GetWindowLongPtrW(hwnd,GWLP_USERDATA);

    menuitem.cbSize=sizeof(MENUITEMINFOW);menuitem.fMask=MIIM_S
TATE;

    menuitem.fState=flags&SF_BLOCKINPUT?MFS_CHECKED:0;
                SetMenuItemInfoW(menu,IDM_LOCKINPUT,0,&menuitem);
                SetWindowLongPtrW(hwnd,GWLP_USERDATA,flags);
            }return 0;
            case
WM_SYSCOMMAND:if((wParam&0xFFF0)==SC_MAXIMIZE){
                MaxScreenViewSize(hwnd);
                return 0;
            }else switch(wParam){
                case IDM_GRAYSCALE:if(HMENU
menu=GetSystemMenu(hwnd,0)){
                    MENUITEMINFOW menuitem;INT_PTR
flags=GetWindowLongPtrW(hwnd,GWLP_USERDATA);
                    flags^=SF_GRAYSCALE;

    menuitem.cbSize=sizeof(MENUITEMINFOW);menuitem.fMask=MIIM_S
TATE;

    menuitem.fState=flags&SF_GRAYSCALE?MFS_CHECKED:0;

    SetMenuItemInfoW(menu,IDM_GRAYSCALE,0,&menuitem);
                    SetWindowLongPtrW(hwnd,GWLP_USERDATA,flags);
                }
                case IDM_REFRESH:
```

```
                    RefreshScreen(hwnd,true);
                    *(uint64_t*)GetPropW(hwnd,L"SampleTime")=0;
                break;
                case IDM_SYNCCLIPBOARD:if(HMENU
menu=GetSystemMenu(hwnd,0)){
                    MENUITEMINFOW menuitem;INT_PTR
flags=GetWindowLongPtrW(hwnd,GWLP_USERDATA);
                    flags^=SF_SYNCCLIP;

    menuitem.cbSize=sizeof(MENUITEMINFOW);menuitem.fMask=MIIM_S
TATE;

    menuitem.fState=flags&SF_SYNCCLIP?MFS_CHECKED:0;

    SetMenuItemInfoW(menu,IDM_SYNCCLIPBOARD,0,&menuitem);
                    SetWindowLongPtrW(hwnd,GWLP_USERDATA,flags);
                }
                break;
                case IDM_LOCKINPUT:if(HMENU
menu=GetSystemMenu(hwnd,0)){
                    INT_PTR
flags=GetWindowLongPtrW(hwnd,GWLP_USERDATA);
                    {MENUITEMINFOW menuitem;
                    flags^=SF_BLOCKINPUT;

    menuitem.cbSize=sizeof(MENUITEMINFOW);menuitem.fMask=MIIM_S
TATE;

    menuitem.fState=flags&SF_BLOCKINPUT?MFS_CHECKED:0;
```

```
SetMenuItemInfoW(menu,IDM_LOCKINPUT,0,&menuitem);

SetWindowLongPtrW(hwnd,GWLP_USERDATA,flags);}


allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT))
                GetDiscordPeerInfo(hwnd,sm.lobby_id,sm.user_id);
                sm.channel=1;

sm.msg.Type=flags&SF_BLOCKINPUT?RDP_MESSAGE::BlockInput:RDP_MESSAGE::UnblockInput;
                sm.msg.Timestamp=GetTimestamp();
                sm.msgsize=RDP_MESSAGE_HEADER_SIZE;

SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm.msg,(LPARAM)&sm);
            }
            break;
            case IDM_RUN:if(HWND edit=GetDlgItem(hwnd,IDC_EDIT))ShowWindow(edit,SW_SHOW);
            else{
                RECT rect;
                if(GetClientRect(hwnd,&rect)){int w=(rect.right-rect.left)/2,h=30*dpiY/defdpiY;
                if(HWND edit=CreateWindowExW(WS_EX_TOOLWINDOW,WC_EDITW,0,WS_CHILD|WS_CLIPSIBLINGS|WS_CLIPCHILDREN|WS_CAPTION|WS_SYSMENU|ES_AUTOHSCROLL,(rect.left+rect.right-w)/2,(rect.top+rect.bottom-h)/2,w*2,h*2,hwnd,(HMENU)IDC_EDIT,0,0)){
```

```
                                WINDOWINFO wi;
                                wi.cbSize=sizeof(WINDOWINFO);

        DefWindowProcW(edit,WM_SETTEXT,0,(LPARAM)L"Run");
                                if(GetWindowInfo(edit,&wi)){
                                        w+=wi.rcWindow.right-
wi.rcWindow.left+wi.rcClient.left-wi.rcClient.right;
                                        h+=wi.rcWindow.bottom-
wi.rcWindow.top+wi.rcClient.top-wi.rcClient.bottom;
                                        SetWindowPos(edit,0,(rect.left+rect.right-
w)/2,(rect.top+rect.bottom-
h)/2,w,h,SWP_DEFERERASE|SWP_FRAMECHANGED|SWP_NOSENDCHAN
GING|SWP_NOREDRAW|SWP_NOOWNERZORDER|SWP_NOZORDER|SWP
_SHOWWINDOW);
                                }
                                SetFocus(edit);

        OldEditWindowProc=(WNDPROC)SetWindowLongPtrW(edit,GWLP_WN
DPROC,(INT_PTR)RunEditWindowProc);
                        }}

                }
                break;
                case IDM_ENDTASK:{

        allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT))
                        GetDiscordPeerInfo(hwnd,sm.lobby_id,sm.user_id);
                        sm.channel=1;
                        sm.msg.Type=RDP_MESSAGE::EndTask;
                        sm.msg.Timestamp=GetTimestamp();
```

```
                    sm.msgsize=RDP_MESSAGE_HEADER_SIZE;

        SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm.msg,(LP
ARAM)&sm);
                }break;

        default:if(IDM_DEVBASE<=wParam&&wParam<0xF000)if(HMENU
menu=(HMENU)GetPropW(hwnd,L"DeviceMenu")){
                    unsigned c=GetMenuItemCount(menu);
                    for(unsigned i=0;i<c;++i){
                        MENUITEMINFOW mi;
                        mi.cbSize=sizeof mi;
                        mi.fMask=MIIM_STATE;
                        mi.fState=i==wParam-
IDM_DEVBASE?MFS_CHECKED:0;
                        SetMenuItemInfoW(menu,i,1,&mi);
                    }
                }
            }break;
            case                                    WM_SIZING:if(UINT_PTR
ScreenSize=(UINT_PTR)GetPropW(hwnd,L"ScreenSize")){
                WINDOWINFO winfo;winfo.cbSize=sizeof(WINDOWINFO);
                if(GetWindowInfo(hwnd,&winfo)){
                    unsigned
cx=LOWORD(ScreenSize)+winfo.rcWindow.right-winfo.rcWindow.left-
winfo.rcClient.right+winfo.rcClient.left;
                    unsigned
cy=HIWORD(ScreenSize)+winfo.rcWindow.bottom-winfo.rcWindow.top-
winfo.rcClient.bottom+winfo.rcClient.top;
                    switch(wParam){
```

```
case WMSZ_LEFT:if(LPRECT(lParam)->right-LPRECT(lParam)->left>cx)LPRECT(lParam)->left=LPRECT(lParam)->right-cx;break;
case WMSZ_RIGHT:if(LPRECT(lParam)->right-LPRECT(lParam)->left>cx)LPRECT(lParam)->right=LPRECT(lParam)->left+cx;break;
case WMSZ_TOPLEFT:if(LPRECT(lParam)->right-LPRECT(lParam)->left>cx)LPRECT(lParam)->left=LPRECT(lParam)->right-cx;
case WMSZ_TOP:top:if(LPRECT(lParam)->bottom-LPRECT(lParam)->top>cy)LPRECT(lParam)->top=LPRECT(lParam)->bottom-cy;break;
case WMSZ_TOPRIGHT:if(LPRECT(lParam)->right-LPRECT(lParam)->left>cx)LPRECT(lParam)->right=LPRECT(lParam)->left+cx;goto top;
case WMSZ_BOTTOMLEFT:if(LPRECT(lParam)->right-LPRECT(lParam)->left>cx)LPRECT(lParam)->left=LPRECT(lParam)->right-cx;
case WMSZ_BOTTOM:bot:if(LPRECT(lParam)->bottom-LPRECT(lParam)->top>cy)LPRECT(lParam)->bottom=LPRECT(lParam)->top+cy;break;
case WMSZ_BOTTOMRIGHT:if(LPRECT(lParam)->right-LPRECT(lParam)->left>cx)LPRECT(lParam)->right=LPRECT(lParam)->left+cx;goto bot;
}
}
}return 1;
case WM_SIZE:if(UINT_PTR ScreenSize=(UINT_PTR)GetPropW(hwnd,L"ScreenSize")){
INT_PTR ViewOrigin=(INT_PTR)GetPropW(hwnd,L"ViewOrigin");int
```

```
viewx=GET_X_LPARAM(ViewOrigin),viewy=GET_Y_LPARAM(ViewOrigin);
BITMAP bmpinfo;
                    if(viewx<0)viewx=0;else
if(viewx>GET_X_LPARAM(ScreenSize)-
GET_X_LPARAM(lParam))viewx=GET_X_LPARAM(ScreenSize)-
GET_X_LPARAM(lParam);
                    if(viewy<0)viewy=0;else
if(viewy>GET_Y_LPARAM(ScreenSize)-
GET_Y_LPARAM(lParam))viewy=GET_Y_LPARAM(ScreenSize)-
GET_Y_LPARAM(lParam);

        if(GET_X_LPARAM(ScreenSize)<=GET_X_LPARAM(lParam))viewx=0;

        if(GET_Y_LPARAM(ScreenSize)<=GET_Y_LPARAM(lParam))viewy=0;
                    if((viewy<<16|viewx)!=ViewOrigin){

        SetPropW(hwnd,L"ViewOrigin",(HANDLE)(viewy<<16|viewx));
                            InvalidateRect(hwnd,0,0);
                    }
                }
                return 0;
                case                        SCR_DEVNAME:if(PRDP_MESSAGE
msg=(PRDP_MESSAGE)lParam)if(HMENU
menu=(HMENU)GetPropW(hwnd,L"DeviceMenu")){
                    unsigned
c=GetMenuItemCount(menu),i=0;MENUITEMINFOA    mi;MENUITEMINFOW
mii;
                    mi.cbSize=sizeof mi;

        mi.fMask=MIIM_ID|MIIM_STATE|MIIM_STRING|MIIM_FTYPE;
```

```
                mi.fType=MFT_RADIOCHECK;
                mii.cbSize=sizeof mii;
                mii.fMask=MIIM_STATE;
                for(char*p=msg->str;p<msg->str+wParam;++i){
                        unsigned len=strlen(p);

        mi.fState=i<c&&GetMenuItemInfoW(menu,i,1,&mii)?mii.fState:0;
                        if(p[len+1]){
                                mi.fState|=MFS_DEFAULT;
                                if(c==0)mi.fState|=MFS_CHECKED;
                        }else mi.fState&=~MFS_DEFAULT;
                        mi.wID=IDM_DEVBASE+i;
                        mi.dwTypeData=p;
                        if(i<c)SetMenuItemInfoA(menu,i,1,&mi);else
InsertMenuItemA(menu,i,1,&mi);
                        p+=len+1+1+sizeof(RECT);
                }
                if(i<c)for(unsigned                                j=c-1;j>=i;--
j)RemoveMenu(menu,j,MF_BYPOSITION);
        }return 0;
        case                        SCR_SETIMAGE:if(PRDP_MESSAGE
msg=(PRDP_MESSAGE)lParam){
                DWORD
ExpectedTimestamp=(UINT_PTR)GetPropW(hwnd,L"ExpectedTimestamp"),Last
Timestamp=(UINT_PTR)GetPropW(hwnd,L"LastTimestamp");
                UINT_PTR
refresh=(UINT_PTR)GetPropW(hwnd,L"RefreshInterval");
                if((msg-
>Timestamp==ExpectedTimestamp||0<refresh&&refresh<INT_MAX&&msg-
```

```
>Timestamp<=ExpectedTimestamp)&&int(msg->Timestamp-
LastTimestamp)>=0){
                        IMFMediaBuffer*mbuf=CreateMFMWBuffer(msg-
>VideoSample.Data,wParam-
offsetof(RDP_MESSAGE::VIDEO_SAMPLE,Data),wParam-
offsetof(RDP_MESSAGE::VIDEO_SAMPLE,Data),0,0,0,msg);
                        if(!mbuf){
                                HeapFree(Heap,0,msg);
                                return 0;
                        }
                        IMFTransform*dec,*conv;
                        bool    ShouldReset=ResetScreenViewSize(hwnd,msg-
>VideoSample.Width,msg->VideoSample.Height);

        if(GetVideoDecoder(hwnd,msg,dec,conv))ShouldReset=true;
                        if(msg-
>Type==RDP_MESSAGE::CameraContinue)ShouldReset=true;
                        if(dec&&conv){
                                IMFSample*sample;DWORD
DecInputStreamID,DecOutputStreamID,ConvInputStreamID,ConvOutputStreamI
D;BITMAP bmpinfo;
                                if(dec-
>GetStreamIDs(1,&DecInputStreamID,1,&DecOutputStreamID))DecInputStreamI
D=DecOutputStreamID=0;
                                if(conv-
>GetStreamIDs(1,&ConvInputStreamID,1,&ConvOutputStreamID))ConvInputStr
eamID=ConvOutputStreamID=0;
                                if(pMFCreateSample(&sample))sample=0;
                                else if(sample->AddBuffer(mbuf)){
                                        sample->Release();
```

```
                                        sample=0;
                                    }
                                    if(sample)if(HBITMAP
bmp=(HBITMAP)GetPropW(hwnd,L"ScreenBitmap"))if(GetObjectW(bmp,sizeof
(BITMAP),&bmpinfo))

        if(IMFMediaBuffer*rgbbuf=CreateMFMWBuffer(bmpinfo.bmBits,bmpinfo.
bmWidthBytes*bmpinfo.bmHeight,bmpinfo.bmWidthBytes*bmpinfo.bmHeight,b
mpinfo.bmWidth,bmpinfo.bmHeight)){
                                        MFT_OUTPUT_DATA_BUFFER
decodb,convodb;DWORD status;

        uint64_t*SampleTime=(uint64_t*)GetPropW(hwnd,L"SampleTime");
                                        sample->SetSampleDuration(10000000);
                                        sample-
>SetSampleTime(SampleTime[0]+=10000000);
                                        sample-
>SetUINT32(MFSampleExtension_CleanPoint,msg->VideoSample.CleanPoint);
                                        HRESULT                    hr=dec-
>ProcessInput(DecInputStreamID,sample,0);
                                        sample->Release();
                                        sample=0;
                                        decodb.pSample=convodb.pSample=0;
                                        MFT_OUTPUT_STREAM_INFO
decosi,convosi;
                                        if(hr==0){
                                            hr=dec-
>GetOutputStreamInfo(DecOutputStreamID,&decosi);
```

```
if(hr==0&&decosi.dwFlags!=MFT_OUTPUT_STREAM_PROVIDES_SAMPLES)decodb.pSample=AllocateSample(decosi.cbSize);

                                        hr=conv->GetOutputStreamInfo(ConvOutputStreamID,&convosi);
                                        if(hr==0)

if(convosi.dwFlags==MFT_OUTPUT_STREAM_PROVIDES_SAMPLES||pMFCreateSample(&convodb.pSample))convodb.pSample=0;
                                        else if(convodb.pSample->AddBuffer(rgbbuf)){
                                                convodb.pSample->Release();
                                                convodb.pSample=0;
                                        }
                                }

if(hr==0&&(decodb.pSample||decosi.dwFlags==MFT_OUTPUT_STREAM_PROVIDES_SAMPLES)&&(convodb.pSample||convosi.dwFlags==MFT_OUTPUT_STREAM_PROVIDES_SAMPLES)){
                                        bool needmore=false;unsigned outcount=0;

        decodb.dwStreamID=DecOutputStreamID;
                                        decodb.dwStatus=status=0;
                                        decodb.pEvents=0;
                                        do{
                                                hr=dec->ProcessOutput(0,1,&decodb,&status);
                                                if(hr==0){
```

```
hasoutput:                                          ++outcount;

        if(decodb.pEvents)decodb.pEvents->Release();
                                                            hr=conv-
>ProcessInput(ConvInputStreamID,decodb.pSample,0);
                                                            if(hr==0){

        convodb.dwStreamID=ConvOutputStreamID;

        convodb.dwStatus=status=0;

        convodb.pEvents=0;
                                                                do{
                                                                    hr=conv-
>ProcessOutput(0,1,&convodb,&status);
                                                                    if(hr)break;

        InvalidateRect(hwnd,0,0);

        if(convodb.pEvents)convodb.pEvents->Release();

        }while(convodb.dwStatus==MFT_OUTPUT_DATA_BUFFER_INCOMPL
ETE);
                                                            }
                                                        }else
if(hr==MF_E_TRANSFORM_NEED_MORE_INPUT){

        if(outcount==0)needmore=true;
```

```
                                                    }else
if(hr==MF_E_TRANSFORM_STREAM_CHANGE&&decodb.dwStatus==MFT_
OUTPUT_DATA_BUFFER_FORMAT_CHANGE){
                                                        IMFMediaType*mt;
                                                        if(dec-
>GetOutputAvailableType(DecOutputStreamID,0,&mt)==0){
                                                            mt-
>SetGUID(MF_MT_SUBTYPE,MFVideoFormat_IYUV);
                                                            dec-
>SetOutputType(DecOutputStreamID,mt,0);
                                                            mt->Release();
                                                            unsigned
oldsize=decosi.cbSize;
                                                            hr=dec-
>GetOutputStreamInfo(DecOutputStreamID,&decosi);

        if(hr==0&&oldsize!=decosi.cbSize){

        if(decodb.pSample){

        decodb.pSample->Release();

        decodb.pSample=0;
                                                                }

        if(decosi.dwFlags!=MFT_OUTPUT_STREAM_PROVIDES_SAMPLES)de
codb.pSample=AllocateSample(decosi.cbSize);
                                                            }
                                                            hr=dec-
>ProcessOutput(0,1,&decodb,&status);
```

```
                                                    if(hr==0)goto
hasoutput;
                                                    else
if(hr==MF_E_TRANSFORM_NEED_MORE_INPUT&&outcount==0)needmore
=true;
                                                        }
                                                }else break;

        }while(decodb.dwStatus==MFT_OUTPUT_DATA_BUFFER_INCOMPLE
TE);
                                            unsigned
outaccum=(UINT_PTR)GetPropW(hwnd,L"OutputCount");
                                            /*if(outaccum+outcount==0&&hr){
                                                wchar_t s[96];
                                                unsigned
n=_snwprintf(s,_countof(s),needmore?L"Received Bytes:%u. Decoder Status:
0x%08X. Need more input.      ":L"Received Bytes:%u. Decoder Status: 0x%08X.
",wParam,hr);
                                                PrintTextToBitmap(hwnd,s,n);
                                                InvalidateRect(hwnd,0,0);
                                            }*/

        if(outcount)SetPropW(hwnd,L"OutputCount",HANDLE(outaccum+outcoun
t));

        if(GetWindowLongPtrW(hwnd,GWL_STYLE)&WS_VISIBLE&&(needmo
re||refresh==0)){
                                                if(msg-
>Type==RDP_MESSAGE::CameraContinue)RefreshCamera(hwnd,false);else
RefreshScreen(hwnd,false);
```

```
                                        }
                                    }
                                    if(convodb.pSample)convodb.pSample-
>Release();
                                    if(decodb.pSample)decodb.pSample-
>Release();
                                    rgbbuf->Release();
                                }
                                if(sample)sample->Release();
                            }
                            if(ShouldReset)MaxScreenViewSize(hwnd);
                            SetPropW(hwnd,L"LastTimestamp",(HANDLE)msg-
>Timestamp);
                            INT_PTR
flags=GetWindowLongPtrW(hwnd,GWLP_USERDATA),newflags=msg-
>Type==RDP_MESSAGE::PrintScreenInputBlocked?flags|SF_BLOCKINPUT:fla
gs&~SF_BLOCKINPUT;

        if(newflags!=flags)SetWindowLongPtrW(hwnd,GWLP_USERDATA,newfl
ags);
                            mbuf->Release();
                        }
                    }return 0;
                    case SCR_RUNERROR:{
                        DWORD
ExpectedTimestamp=(UINT_PTR)GetPropW(hwnd,L"ExpectedTimestamp1"),La
stTimestamp=(UINT_PTR)GetPropW(hwnd,L"LastTimestamp1");
                        if(wParam==ExpectedTimestamp&&int(wParam-
LastTimestamp)>=0){
                            wchar_t s[64];
```

```
                unsigned
n=FormatMessageW(FORMAT_MESSAGE_FROM_SYSTEM,0,lParam,0,s,_cou
ntof(s),0);
                while(n&&(s[n-1]==L'\r'||s[n-1]==L'\n'))--n;

        SetPropW(hwnd,L"LastTimestamp1",(HANDLE)wParam);
                MessageBoxW(hwnd,s,0,MB_ICONERROR);
            }
        }
        return 0;
        case SCR_GETCLIP:{
            bool opened=OpenClipboard(hwnd);
            HGLOBAL hMem=opened?GetClipboardData(wParam):0;
            unsigned size=hMem?GlobalSize(hMem):0;
            if(PSENDMSGSTRUCT
sm=(PSENDMSGSTRUCT)HeapAlloc(Heap,0,sizeof(SENDMSGSTRUCT)+size
of(uint32_t)+size)){
                if(size)if(void*p=GlobalLock(hMem)){
;                   memmove(sm->msg.u+1,p,size);
                    GlobalUnlock(hMem);
                    CloseClipboard();
                    opened=false;

        if(size>GetPrivateProfileIntW(L"MASTER",L"ClipboardSizeThreashold",1
6384,INIFileName)){
                        wchar_t s[96];
                        _snwprintf(s,_countof(s),L"The size of
clipboard data is %u bytes. Still want to send?",size);
```

```
        if(MessageBoxW(hwnd,s,L"Clipboard",MB_ICONQUESTION|MB_YESNO)==IDNO)size=-4;
                                        }
                                }else size=0;
                                GetDiscordPeerInfo(hwnd,sm->lobby_id,sm->user_id);
                                sm->msg.Type=RDP_MESSAGE::SendClipboardData;
                                sm->msg.u[0]=wParam;
                                sm->msgsize=int(size)<0?RDP_MESSAGE_HEADER_SIZE:RDP_MESSAGE_HEADER_SIZE+sizeof(uint32_t)+size;
                                sm->channel=1;
                                sm->msg.Timestamp=GetTimestamp();

        SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm->msg,(LPARAM)sm);
                                HeapFree(Heap,0,sm);
                        }
                        if(opened)CloseClipboard();
                }return 0;
        }
        return DefWindowProcW(hwnd,uMsg,wParam,lParam);
}

HWND FindScreenViewerWindow(int64_t user_id)
{
        wchar_t s[64];
        s[0]=L'S';s[1]=L'C';s[2]=L'R';s[3]=L'W';s[4]=L'N';s[5]=L'D';
        _ui64tow(user_id,s+6,36);
        HWND hwnd=(HWND)GetPropW(MainWindow,s);DWORD pid;
```

```
	return
IsWindow(hwnd)&&GetWindowThreadProcessId(hwnd,&pid)&&pid==GetCurre
ntProcessId()&&ValidateScreenViewerWindow(hwnd,0,user_id)?hwnd:0;
}

static LRESULT CALLBACK MessageProc(int nCode,WPARAM
wParam,LPARAM lParam)
{
	if(nCode==MSGF_MENU&&(PMSG(lParam)-
>message==WM_MOUSEMOVE||PMSG(lParam)-
>message==WM_LBUTTONDOWN||PMSG(lParam)-
>message==WM_LBUTTONUP)){
		HWND tb=WindowFromPoint(PMSG(lParam)->pt);wchar_t s[256];

	if(GetWindowThreadProcessId(tb,0)==GetCurrentThreadId()&&GetClassN
ameW(tb,s,_countof(s))==_countof(TRACKBAR_CLASSW)-
1&&_wcsicmp(s,TRACKBAR_CLASSW)==0){
			if(PMSG(lParam)-
>message==WM_MOUSEMOVE&&PMSG(lParam)-
>wParam&MK_LBUTTON||PMSG(lParam)-
>message==WM_LBUTTONDOWN){
				int
min=SendMessageW(tb,TBM_GETRANGEMIN,0,0),max=SendMessageW(tb,TB
M_GETRANGEMAX,0,0);RECT rect;

	SendMessageW(tb,TBM_GETCHANNELRECT,0,(LPARAM)&rect);
				ScreenToClient(tb,&PMSG(lParam)->pt);
				if(max>min&&rect.right-rect.left){
					int	pos=(PMSG(lParam)->pt.x-rect.left)*(max-
min)/(rect.right-rect.left);
```

```
                                if(pos<min)pos=min;else if(pos>max)pos=max;
                                SendMessageW(tb,TBM_SETPOS,1,pos);
                                if(HWND owner=GetWindow(tb,GW_OWNER)){
                                        INT_PTR
i=GetWindowLongPtrW(tb,GWLP_USERDATA);

        if(i)SetPropW(owner,L"Quality",HANDLE(pos));else{

        SetPropW(owner,L"RefreshInterval",HANDLE(pos>=max?UINT_MAX:un
signed(pos*TimerTick)));

        if(pos&&pos<max)SetTimer(owner,1,unsigned(pos*TimerTick),0);else
KillTimer(owner,1);
                                                if(pos==0)RefreshScreen(owner,false);
                                        }
                                        if(HMENU
menu=GetWindowLongPtrW(owner,GWLP_WNDPROC)==(INT_PTR)ScreenVi
ewerWindowProc?GetSystemMenu(owner,0):(HMENU)GetPropW(owner,L"Cont
extMenu")){
                                                wchar_t     s[64];MENUITEMINFOW
mi;

        mi.cbSize=sizeof(MENUITEMINFOW);
                                                mi.fMask=MIIM_STRING;
                                                mi.dwTypeData=s;

        if(i)_snwprintf(s,_countof(s),L"Quality: %u",pos);else

        if(pos==0)mi.dwTypeData=(PWSTR)L"Refresh: Auto";
```

```
                                                    else
if(pos<max)_snwprintf(s,_countof(s),L"Refresh:
%ums",unsigned(pos*TimerTick));
                                                    else
mi.dwTypeData=(PWSTR)L"Refresh: Manual";
                                                    SetMenuItemInfoW(menu,i,1,&mi);
                                                }
                                            }
                                        }
                                    }
                                    return 1;
                                }
                            }
                            return CallNextHookEx(0,nCode,wParam,lParam);
}

typedef struct _SCREENVIEWPARAMS{
    int64_t lobby_id,user_id;HICON icon;long refcount;unsigned dim;HWND
hwnd;HANDLE hEvent;wchar_t title[0];
}SCREENVIEWPARAMS,*PSCREENVIEWPARAMS;

static DWORD CALLBACK
ScreenViewerThreadProc(PSCREENVIEWPARAMS params)
{
    CoInitializeEx(0,COINIT_APARTMENTTHREADED|COINIT_DISABLE
_OLE1DDE);
    HWND hwnd=params-
>hwnd=CreateWindowExW(WS_EX_CLIENTEDGE,WC_EDITW,0,WS_CLIPC
HILDREN|WS_OVERLAPPEDWINDOW|ES_READONLY|ES_MULTILINE,C
```

```
W_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAU
LT,0,0,0,0);
	uint64_t SampleTime=0;
	InterlockedIncrement(&params->refcount);
	SetEvent(params->hEvent);
	if(hwnd){
		MSG msg;BOOL bRet;

	OldEditWindowProc=(WNDPROC)SetWindowLongPtrW(hwnd,GWLP_W
NDPROC,(INT_PTR)ScreenViewerWindowProc);
		SetDiscordPeerInfo(hwnd,params->lobby_id,params->user_id);
		SetPropW(hwnd,L"AvatarIcon",(HANDLE)params->icon);
		unsigned
Quality=GetPrivateProfileIntW(L"MASTER",L"ScreenViewQuality",0,INIFileNa
me),SizeClamp=GetPrivateProfileIntW(L"MASTER",L"ScreenViewSizeClamp",0
,INIFileName);
		if(Quality>100)Quality=100;
		int
gray=GetPrivateProfileIntW(L"MASTER",L"ScreenViewGrayscale",0,INIFileNa
me)?SF_GRAYSCALE:0,syncclip=GetPrivateProfileIntW(L"MASTER",L"Screen
ViewSyncClipboard",0,INIFileName)?SF_SYNCCLIP:0;
		unsigned
refresh=GetPrivateProfileIntW(L"MASTER",L"ScreenViewRefreshInterval",0,INI
FileName);
		SetPropW(hwnd,L"Quality",(HANDLE)Quality);
		SetPropW(hwnd,L"SizeClamp",(HANDLE)SizeClamp);
		SetPropW(hwnd,L"RefreshInterval",(HANDLE)refresh);
		SetWindowLongPtrW(hwnd,GWLP_USERDATA,gray|syncclip);
		SetPropW(hwnd,L"SampleTime",&SampleTime);
```

```
DefWindowProcW(hwnd,WM_SETICON,ICON_SMALL,(LPARAM)params->icon);

DefWindowProcW(hwnd,WM_SETICON,ICON_BIG,(LPARAM)params->icon);
        ResetScreenViewSize(hwnd,LOWORD(params->dim),HIWORD(params->dim));
        MaxScreenViewSize(hwnd);
        RefreshScreen(hwnd,true);
        static HMENU MonitorMenu;
        if(!EmptyMenu)EmptyMenu=CreatePopupMenu();
        if(HMENU menu=GetSystemMenu(hwnd,0)){
            wchar_t s[64];
            MENUITEMINFOW menuitems[]={{sizeof(MENUITEMINFOW),MIIM_ID|MIIM_STRING,0,0,IDM_REFRESH,0,0,0,0,s},

    {sizeof(MENUITEMINFOW),MIIM_ID|MIIM_STRING|MIIM_SUBMENU,0,0,IDM_CAMDEVICE,0,0,0,0,(wchar_t*)L"Monitor"},

    {sizeof(MENUITEMINFOW),MIIM_ID|MIIM_STRING|MIIM_SUBMENU,0,0,IDM_QUALITY,EmptyMenu,0,0,0,s},

    {sizeof(MENUITEMINFOW),MIIM_ID|MIIM_STRING|MIIM_STATE,0,unsigned(gray?MFS_CHECKED:0),IDM_GRAYSCALE,0,0,0,0,(wchar_t*)L"Grayscale"},

    {sizeof(MENUITEMINFOW),MIIM_ID|MIIM_STRING|MIIM_STATE,0,
```

```
unsigned(syncclip?MFS_CHECKED:0),IDM_SYNCCLIPBOARD,0,0,0,0,(wchar_t*)L"Sync Clipboard"},

	{sizeof(MENUITEMINFOW),MIIM_ID|MIIM_STRING,0,0,IDM_LOCKINPUT,0,0,0,0,(wchar_t*)L"Lock Input"},

	{sizeof(MENUITEMINFOW),MIIM_ID|MIIM_STRING,0,0,IDM_RUN,0,0,0,0,(wchar_t*)L"Run"},

	{sizeof(MENUITEMINFOW),MIIM_ID|MIIM_STRING,0,0,IDM_ENDTASK,0,0,0,0,(wchar_t*)L"End Task"},

	{sizeof(MENUITEMINFOW),MIIM_ID|MIIM_TYPE,MFT_SEPARATOR,0,IDM_SEPARATOR,0,0,0,0,0}};

	SetPropW(hwnd,L"DeviceMenu",menuitems[1].hSubMenu=CreatePopupMenu());
			for(int i=0;i<_countof(menuitems);++i){
				if(i==0){
					if(refresh==0)wcscpy(s,L"Refresh: Auto");
					else if(refresh<INT_MAX)_snwprintf(s,_countof(s),L"Refresh: %ums",refresh);
					else wcscpy(s,L"Refresh: Manual");
				}else	if(i==2)_snwprintf(s,_countof(s),L"Quality: %u",Quality);
				InsertMenuItemW(menu,i,1,menuitems+i);
			}
		}
		//ShowWindow(hwnd,SW_SHOW);
		MoveWindowToCursor(hwnd,true);
```

```
        while(PeekMessageW(&msg,0,0,0,PM_REMOVE))DispatchMessageW(&
msg);
                DefWindowProcW(hwnd,WM_SETTEXT,0,(LPARAM)params-
>title);
                if(InterlockedDecrement(&params-
>refcount)==0)HeapFree(Heap,0,params),params=0;
                HHOOK
hHook=SetWindowsHookExW(WH_MSGFILTER,MessageProc,0,GetCurrentThr
eadId());
                while((bRet=GetMessageW(&msg,0,0,0))&&bRet!=-1){
                        TranslateMessage(&msg);
                        DispatchMessageW(&msg);
                }
                UnhookWindowsHookEx(hHook);
        }
        if(params&&InterlockedDecrement(&params-
>refcount)==0)HeapFree(Heap,0,params);
        CoUninitialize();
        return 0;
}

void ShowScreenViewerWindow(int64_t lobby_id,int64_t user_id,const
wchar_t*username,HICON icon)
{
        if(HWND hwnd=FindScreenViewerWindow(user_id)){
                //ShowWindow(hwnd,SW_SHOW);
                MoveWindowToCursor(hwnd,true);
                if(icon)DestroyIcon(icon);
                return;
```

```
        }
        unsigned namelen=wcslen(username);
        if(!CrossCursor)CrossCursor=(HCURSOR)LoadImageW(0,IDC_CROSS,I
MAGE_CURSOR,0,0,LR_DEFAULTSIZE|LR_SHARED);
        if(!SizeCursor)SizeCursor=(HCURSOR)LoadImageW(0,IDC_SIZEALL,IM
AGE_CURSOR,0,0,LR_DEFAULTSIZE|LR_SHARED);
        ScreenViewMouseMoveInterval=GetPrivateProfileIntW(L"MASTER",L"Sc
reenViewMouseMoveInterval",128,INIFileName);
        if(PSCREENVIEWPARAMS
params=(PSCREENVIEWPARAMS)HeapAlloc(Heap,0,sizeof(SCREENVIEWP
ARAMS)+(sizeof"Screen - "+namelen+1)*sizeof(wchar_t))){
      params->refcount=1;
              if(params-
>hEvent=CreateEventExW(0,0,0,SYNCHRONIZE|EVENT_MODIFY_STATE)){
                      wchar_t s[64];
                      s[0]=L’S’;s[1]=L’C’;s[2]=L’R’;s[3]=L’E’;s[4]=L’S’;
                      _ui64tow(user_id,s+5,36);
                      params->lobby_id=lobby_id;
                      params->user_id=user_id;
                      params->icon=icon;
                      params->dim=(UINT_PTR)GetPropW(MainWindow,s);
                      _snwprintf(params->title,sizeof"Screen                -
"+namelen+1,L"Screen - %s",username);
                      if(HANDLE
hThread=CreateThread(0,0,(LPTHREAD_START_ROUTINE)ScreenViewerThre
adProc,params,0,0)){

      s[0]=L’S’;s[1]=L’C’;s[2]=L’R’;s[3]=L’W’;s[4]=L’N’;s[5]=L’D’;
                          _ui64tow(user_id,s+6,36);
                          WaitForSingleObject(params->hEvent,INFINITE);
```

```
                        SetPropW(MainWindow,s,params->hwnd);
                    }
                    CloseHandle(params->hEvent);
                }
                if(InterlockedDecrement(&params-
>refcount)==0)HeapFree(Heap,0,params);
        }
}

static void GetSourceFormats(HWND hwnd,int channel=1,int di=-1)
{
        if(di<0)di=(INT_PTR)GetPropW(hwnd,L"DeviceIndex");
        allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT)+sizeof(uint3
2_t))
        GetDiscordPeerInfo(hwnd,sm.lobby_id,sm.user_id);
        sm.channel=channel;
        sm.msg.Type=RDP_MESSAGE::GetVideoSourceFormat;
        sm.msgsize=RDP_MESSAGE_HEADER_SIZE+sizeof(uint32_t);
        sm.msg.u[0]=di;
        sm.msg.Timestamp=GetTimestamp();
        SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm.msg,(LP
ARAM)&sm);
        SetPropW(hwnd,L"ExpectedTimestamp2",(HANDLE)sm.msg.Timestamp);
}

LRESULT CALLBACK CameraViewerWindowProc(HWND hwnd,UINT
uMsg,WPARAM wParam,LPARAM lParam)
{
        switch(uMsg){
```

```
        case WM_NCPAINT:return OldEditWindowProc(hwnd,uMsg,wParam,lParam);
        case WM_PAINT:if(HBITMAP bmp=(HBITMAP)GetPropW(hwnd,L"ScreenBitmap")){
            PAINTSTRUCT ps;
            if(HDC dc=BeginPaint(hwnd,&ps)){
                if(HDC memdc=CreateCompatibleDC(dc)){
                    HGDIOBJ defbmp=SelectObject(memdc,bmp);

    BitBlt(dc,ps.rcPaint.left,ps.rcPaint.top,ps.rcPaint.right-ps.rcPaint.left,ps.rcPaint.bottom-ps.rcPaint.top,memdc,ps.rcPaint.left,ps.rcPaint.top,SRCCOPY);
                    SelectObject(memdc,defbmp);
                    DeleteDC(memdc);
                }
                EndPaint(hwnd,&ps);
                return 0;
            }
        }break;
        case WM_ERASEBKGND:return 1;
        case WM_CLOSE:if(lParam==‘quit’)DestroyWindow(hwnd);else ShowWindow(hwnd,SW_HIDE);return 0;
        case WM_DESTROY:PostQuitMessage(0);
        if(HICON icon=(HICON)RemovePropW(hwnd,L"AvatarIcon"))DestroyIcon(icon);
        if(HBITMAP bmp=(HBITMAP)RemovePropW(hwnd,L"ScreenBitmap"))DeleteObject(bmp);

    if(IMFTransform*dec=(IMFTransform*)RemovePropW(hwnd,L"VideoDecoder"))dec->Release();
```

```
        if(IMFTransform*conv=(IMFTransform*)RemovePropW(hwnd,L"ColorCo
nverter"))conv->Release();
                if(HMENU
menu=(HMENU)RemovePropW(hwnd,L"DeviceMenu"))DestroyMenu(menu);
                if(HMENU
menu=(HMENU)RemovePropW(hwnd,L"ResolutionMenu"))DestroyMenu(menu)
;
                if(HMENU
menu=(HMENU)RemovePropW(hwnd,L"ContextMenu"))DestroyMenu(menu);
                goto closetb;
                return 0;
                case WM_EXITSIZEMOVE:MaxScreenViewSize(hwnd);
                return 0;
                case                                    WM_MENUSELECT:if(HMENU
menu=(HMENU)GetPropW(hwnd,L"ContextMenu")){
                        RECT rect;

        if(LOWORD(wParam)==3&&HIWORD(wParam)&MF_MOUSESELECT
&&HIWORD(wParam)&MF_POPUP&&lParam==(LPARAM)menu){
                                if(GetMenuItemRect(0,(HMENU)lParam,3,&rect))
                                if(HWND
tb=(HWND)GetPropW(hwnd,L"QualityTrackBar"));
                                else                                    if(HWND
tb=CreateWindowExW(WS_EX_TOPMOST|WS_EX_NOACTIVATE,TRACKB
AR_CLASSW,0,WS_POPUP|WS_CLIPCHILDREN|WS_CLIPSIBLINGS|WS_B
ORDER|TBS_HORZ|TBS_NOTICKS,

        rect.right+3*dpiX/defdpiX,rect.top,rect.right-
rect.left+64*dpiX/defdpiX,rect.bottom-rect.top,hwnd,0,0,0)){
```

```
SendMessageW(tb,TBM_SETRANGE,0,100<<16|0);
                    SetWindowLongPtrW(tb,GWLP_USERDATA,3);
                    UINT_PTR
Quality=(UINT_PTR)GetPropW(hwnd,L"Quality");
                    SendMessageW(tb,TBM_SETPOS,1,Quality);
                    ShowWindow(tb,SW_SHOWNA);
                    SetPropW(hwnd,L"QualityTrackBar",tb);
                }
            }else                                   if(HWND
tb=(HWND)RemovePropW(hwnd,L"QualityTrackBar"))DestroyWindow(tb);
        }
        return 0;
        case
WM_CONTEXTMENU:if(wParam==(WPARAM)hwnd&&DefWindowProcW(h
wnd,WM_NCHITTEST,0,lParam)==HTCLIENT)if(HMENU
menu=(HMENU)GetPropW(hwnd,L"ContextMenu")){

    allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT))
        GetDiscordPeerInfo(hwnd,sm.lobby_id,sm.user_id);
        sm.channel=0;
        sm.msg.Type=RDP_MESSAGE::GetVideoSource;
        sm.msgsize=RDP_MESSAGE_HEADER_SIZE;
        sm.msg.Timestamp=GetTimestamp();

    SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm.msg,(LP
ARAM)&sm);

    SetPropW(hwnd,L"ExpectedTimestamp1",(HANDLE)sm.msg.Timestamp);
        GetSourceFormats(hwnd,0);
```

```
int
sel=TrackPopupMenuEx(menu,TPM_NONOTIFY|TPM_RETURNCMD|TPM_RI
GHTBUTTON,GET_X_LPARAM(lParam),GET_Y_LPARAM(lParam),hwnd,0);
wchar_t s[96];MENUITEMINFOW
mi;mi.cbSize=sizeof(MENUITEMINFOW);
if(sel==IDM_REFRESH){
UINT_PTR
refresh=(UINT_PTR)GetPropW(hwnd,L"RefreshInterval");
refresh=refresh?0:UINT_MAX;
if(refresh==0)RefreshCamera(hwnd,false);
SetPropW(hwnd,L"RefreshInterval",(HANDLE)refresh);
mi.fMask=MIIM_STATE;
mi.fState=refresh?0:MFS_CHECKED;
SetMenuItemInfoW(menu,sel,0,&mi);
}else
if(IDM_DEVBASE<=sel&&sel<IDM_DEVBASE+1000){
if(HMENU
devmenu=(HMENU)GetPropW(hwnd,L"DeviceMenu")){
mi.fMask=MIIM_STATE;
unsigned i=GetMenuItemCount(devmenu);
while(i-->0){
mi.fState=i==sel-
IDM_DEVBASE?MFS_CHECKED:0;
SetMenuItemInfoW(devmenu,i,1,&mi);
}
unsigned
oi=(UINT_PTR)GetPropW(hwnd,L"DeviceIndex");
SetPropW(hwnd,L"DeviceIndex",HANDLE(sel-
IDM_DEVBASE));
```

```
                                if(oi!=sel-
IDM_DEVBASE)GetSourceFormats(hwnd,1);
                                //RefreshCamera(hwnd,true);
                            }
                        }else if(IDM_RESBASE<=sel&&sel<IDM_RESBASE+1000){
                            if(HMENU
resmenu=(HMENU)GetPropW(hwnd,L"ResolutionMenu")){
                                unsigned cx,cy;
                                mi.fMask=MIIM_STRING;
                                mi.dwTypeData=s;
                                mi.cch=_countof(s);

        if(GetMenuItemInfoW(resmenu,sel,0,&mi)&&swscanf(s,L"%u×%u",&cx,&
cy)==2){
                                        mi.fMask=MIIM_STATE;
                                        unsigned i=GetMenuItemCount(resmenu);
                                        while(i-->0){
                                            mi.fState=i==sel-
IDM_RESBASE?MFS_CHECKED:0;

        SetMenuItemInfoW(resmenu,i,1,&mi);
                                        }

        SetPropW(hwnd,L"Resolution",HANDLE(cy<<16|cx));
                                        //RefreshCamera(hwnd,true);
                                }
                            }
                        }
                        return 0;
                    }break;
```

```
		case	WM_LBUTTONDBLCLK:RefreshCamera(hwnd,false,0);return
0;
		case
WM_SETCURSOR:if((HWND)wParam==hwnd&&LOWORD(lParam)==HTCLI
ENT)SetCursor(ArrowCursor);else break;return 1;
		case WM_SHOWWINDOW:if(wParam){

	allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT))
			GetDiscordPeerInfo(hwnd,sm.lobby_id,sm.user_id);
			sm.channel=GetPropW(hwnd,L"Resolution")?0:1;
			sm.msg.Type=RDP_MESSAGE::GetVideoSource;
			sm.msgsize=RDP_MESSAGE_HEADER_SIZE;
			sm.msg.Timestamp=GetTimestamp();

	SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm.msg,(LP
ARAM)&sm);

	SetPropW(hwnd,L"ExpectedTimestamp1",(HANDLE)sm.msg.Timestamp);
			*(uint64_t*)GetPropW(hwnd,L"SampleTime")=0;
			GetSourceFormats(hwnd,sm.channel);

	if(GetPropW(hwnd,L"Resolution"))RefreshCamera(hwnd,false);
			if(UINT_PTR
refresh=(UINT_PTR)GetPropW(hwnd,L"RefreshInterval"))if(refresh<INT_MAX)
SetTimer(hwnd,1,refresh,0);
		}else{
			KillTimer(hwnd,1);

	allocvar(SENDMSGSTRUCT,sm,sizeof(SENDMSGSTRUCT))
			GetDiscordPeerInfo(hwnd,sm.lobby_id,sm.user_id);
```

```
                sm.channel=1;
                sm.msg.Type=RDP_MESSAGE::CameraStop;
                sm.msgsize=RDP_MESSAGE_HEADER_SIZE;
                sm.msg.Timestamp=GetTimestamp();

        SendMessageW(MainWindow,SM_SENDMSG,(WPARAM)&sm.msg,(LP
ARAM)&sm);
            }
closetb:if(HWND
tb=(HWND)RemovePropW(hwnd,L"QualityTrackBar"))DestroyWindow(tb);if(H
WND
tb=(HWND)RemovePropW(hwnd,L"RefreshTrackBar"))DestroyWindow(tb);brea
k;
            case
WM_TIMER:if(wParam==1)if(GetWindowLongPtrW(hwnd,GWL_STYLE)&WS
_VISIBLE){
                RefreshCamera(hwnd,false,0);
            }else KillTimer(hwnd,1);
            return 0;
            case                                    CM_DEVNAME:if(HMENU
menu=(HMENU)GetPropW(hwnd,L"DeviceMenu")){
                PRDP_MESSAGE msg=(PRDP_MESSAGE)lParam;
                int
i=GetMenuItemCount(menu),di=(INT_PTR)GetPropW(hwnd,L"DeviceIndex");
                while(i-->0)RemoveMenu(menu,i,MF_BYPOSITION);

        if(wchar_t*s=(wchar_t*)HeapAlloc(Heap,0,(wParam+1)*sizeof(wchar_t))){
                    unsigned    n=MultiByteToWideChar(CP_UTF8,0,msg-
>str,wParam,s,wParam),l;
                    MENUITEMINFOW mi;
```

```
                        mi.cbSize=sizeof(MENUITEMINFOW);

        mi.fMask=MIIM_ID|MIIM_STRING|MIIM_FTYPE|MIIM_STATE;
                        mi.fType=MFT_RADIOCHECK;

        for(mi.dwTypeData=s,i=0;mi.dwTypeData<s+n;mi.dwTypeData+=l+1,++i)
{
                                l=wcslen(mi.dwTypeData);
                                mi.fState=i==di?MFS_CHECKED:0;
                                mi.wID=IDM_DEVBASE+i;
                                InsertMenuItemW(menu,i,1,&mi);
                        }
                        HeapFree(Heap,0,s);
                }
        }HeapFree(Heap,0,(void*)lParam);
        return 0;
        case                    CM_DEVFMT:if(PRDP_MESSAGE(lParam)-
>u[0]==(UINT_PTR)GetPropW(hwnd,L"DeviceIndex"))if(HMENU
menu=(HMENU)GetPropW(hwnd,L"ResolutionMenu")){
                PRDP_MESSAGE msg=(PRDP_MESSAGE)lParam;
                int i=GetMenuItemCount(menu);
                while(i-->0)RemoveMenu(menu,i,MF_BYPOSITION);
                UINT_PTR
curres=(UINT_PTR)GetPropW(hwnd,L"Resolution");
                if(!curres&&wParam>=sizeof(uint32_t)*2){
                        curres=msg->u[1];
                        SetPropW(hwnd,L"Resolution",(HANDLE)curres);

        ResetScreenViewSize(hwnd,LOWORD(curres),HIWORD(curres));
                        MaxScreenViewSize(hwnd);
```

```
                }
                MENUITEMINFOW mi;wchar_t s[64];
                mi.cbSize=sizeof(MENUITEMINFOW);

        mi.fMask=MIIM_ID|MIIM_STRING|MIIM_FTYPE|MIIM_STATE;
                mi.fType=MFT_RADIOCHECK;
                mi.dwTypeData=s;
                for(i=0;(3+i)*sizeof(uint32_t)*3<=wParam;++i){
                        mi.wID=IDM_RESBASE+i;
                        _snwprintf(s,_countof(s),L"%u×%u",LOWORD(msg-
>u[2+i]),HIWORD(msg->u[2+i]));
                        mi.fState=msg->u[2+i]==curres?MFS_CHECKED:0;
                        InsertMenuItemW(menu,i,1,&mi);
                }
        }HeapFree(Heap,0,(void*)lParam);
        return 0;
        case                SCR_SETIMAGE:if(wParam&&lParam)return
ScreenViewerWindowProc(hwnd,uMsg,wParam,lParam);
        else if(GetPropW(hwnd,L"VideoDecoder")==0){
                int
devcount=GetMenuItemCount((HMENU)GetPropW(hwnd,L"DeviceMenu"));
                PrintTextToBitmap(hwnd,devcount>0?L"Unable to access
video capture device.":L"No video capture device
found.",devcount>0?sizeof"Unable to access video capture device":sizeof"No
video capture device found");
                InvalidateRect(hwnd,0,0);
        }
        return 0;
    }
    return DefWindowProcW(hwnd,uMsg,wParam,lParam);
```

```
}

static DWORD CALLBACK CameraViewerThreadProc(PSCREENVIEWPARAMS params)
{
	CoInitializeEx(0,COINIT_APARTMENTTHREADED|COINIT_DISABLE_OLE1DDE);
	//pMFStartup(MF_VERSION,MFSTARTUP_LITE);
	HWND hwnd=params->hwnd=CreateWindowExW(WS_EX_CLIENTEDGE,WC_EDITW,0,WS_CLIPCHILDREN|WS_CAPTION|WS_SYSMENU|ES_READONLY|ES_MULTILINE,CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,0,0,0,0);
	uint64_t SampleTime=0;
	InterlockedIncrement(&params->refcount);
	SetEvent(params->hEvent);
	if(hwnd){
		MSG msg;BOOL bRet;

	OldEditWindowProc=(WNDPROC)SetWindowLongPtrW(hwnd,GWLP_WNDPROC,(INT_PTR)CameraViewerWindowProc);
		SetDiscordPeerInfo(hwnd,params->lobby_id,params->user_id);
		SetPropW(hwnd,L"AvatarIcon",(HANDLE)params->icon);
		unsigned Quality=GetPrivateProfileIntW(L"MASTER",L"ScreenViewQuality",0,INIFileName);
		if(Quality>100)Quality=100;
		SetPropW(hwnd,L"Quality",(HANDLE)Quality);
		SetPropW(hwnd,L"SampleTime",&SampleTime);
```

```
DefWindowProcW(hwnd,WM_SETICON,ICON_SMALL,(LPARAM)params->icon);

DefWindowProcW(hwnd,WM_SETICON,ICON_BIG,(LPARAM)params->icon);
        ResetScreenViewSize(hwnd,LOWORD(params->dim),HIWORD(params->dim));
        MaxScreenViewSize(hwnd);
        RefreshCamera(hwnd,true);
        if(!EmptyMenu)EmptyMenu=CreatePopupMenu();
        if(HMENU menu=CreatePopupMenu()){
            wchar_t s[64];
            MENUITEMINFOW menuitems[]={{sizeof(MENUITEMINFOW),MIIM_ID|MIIM_STRING|MIIM_SUBMENU,0,0,IDM_CAMDEVICE,CreatePopupMenu(),0,0,0,(wchar_t*)L"Device"},

{sizeof(MENUITEMINFOW),MIIM_ID|MIIM_STRING|MIIM_SUBMENU,0,0,IDM_CAMDEVICE,CreatePopupMenu(),0,0,0,(wchar_t*)L"Resolution"},

{sizeof(MENUITEMINFOW),MIIM_ID|MIIM_STRING|MIIM_STATE,0,MFS_CHECKED,IDM_REFRESH,0,0,0,0,(wchar_t*)L"Auto Refresh"},

{sizeof(MENUITEMINFOW),MIIM_ID|MIIM_STRING|MIIM_SUBMENU,0,0,IDM_QUALITY,EmptyMenu,0,0,0,s}};
            for(int i=0;i<_countof(menuitems);++i){
                if(i==3)_snwprintf(s,_countof(s),L"Quality: %u",Quality);
                InsertMenuItemW(menu,i,1,menuitems+i);
```

```
                }
                SetPropW(hwnd,L"DeviceMenu",menuitems[0].hSubMenu);

        SetPropW(hwnd,L"ResolutionMenu",menuitems[1].hSubMenu);
                SetPropW(hwnd,L"ContextMenu",menu);
            }
            MoveWindowToCursor(hwnd,true);

        while(PeekMessageW(&msg,0,0,0,PM_REMOVE))DispatchMessageW(&
msg);
            DefWindowProcW(hwnd,WM_SETTEXT,0,(LPARAM)params-
>title);
            if(InterlockedDecrement(&params-
>refcount)==0)HeapFree(Heap,0,params),params=0;
            HHOOK
hHook=SetWindowsHookExW(WH_MSGFILTER,MessageProc,0,GetCurrentThr
eadId());
            while((bRet=GetMessageW(&msg,0,0,0))&&bRet!=-1){
                TranslateMessage(&msg);
                DispatchMessageW(&msg);
            }
            UnhookWindowsHookEx(hHook);
        }
        if(params&&InterlockedDecrement(&params-
>refcount)==0)HeapFree(Heap,0,params);
        CoUninitialize();
        return 0;
}
```

```
bool ValidateCameraViewerWindow(HWND hwnd,int64_t lobby_id,int64_t user_id)
{
	return ValidateSlaveWindow(hwnd,lobby_id,user_id,CameraViewerWindowProc);
}

HWND FindCameraViewerWindow(int64_t user_id)
{
	wchar_t s[64];
	s[0]=L’C’;s[1]=L’A’;s[2]=L’M’;s[3]=L’W’;s[4]=L’N’;s[5]=L’D’;
	_ui64tow(user_id,s+6,36);
	HWND hwnd=(HWND)GetPropW(MainWindow,s);DWORD pid;
	return IsWindow(hwnd)&&GetWindowThreadProcessId(hwnd,&pid)&&pid==GetCurrentProcessId()&&ValidateCameraViewerWindow(hwnd,0,user_id)?hwnd:0;
}

void ShowCameraViewerWindow(int64_t lobby_id,int64_t user_id,const wchar_t*username,HICON icon)
{
	if(HWND hwnd=FindCameraViewerWindow(user_id)){
		MoveWindowToCursor(hwnd,true);
		if(icon)DestroyIcon(icon);
		return;
	}
	unsigned namelen=wcslen(username);
	if(PSCREENVIEWPARAMS params=(PSCREENVIEWPARAMS)HeapAlloc(Heap,0,sizeof(SCREENVIEWPARAMS)+(sizeof"Camera - "+namelen+1)*sizeof(wchar_t))){
```

```
            params->refcount=1;
            if(params-
>hEvent=CreateEventExW(0,0,0,SYNCHRONIZE|EVENT_MODIFY_STATE)){
                wchar_t s[64];
                s[0]=L’C’;s[1]=L’M’;s[2]=L’R’;s[3]=L’E’;s[4]=L’S’;
                _ui64tow(user_id,s+5,36);
                params->lobby_id=lobby_id;
                params->user_id=user_id;
                params->icon=icon;
                params->dim=(UINT_PTR)GetPropW(MainWindow,s);
                if(params->dim==0)params->dim=480<<16|640;
                _snwprintf(params->title,sizeof"Camera                -
"+namelen+1,L"Camera - %s",username);
                if(HANDLE
hThread=CreateThread(0,0,(LPTHREAD_START_ROUTINE)CameraViewerThr
eadProc,params,0,0)){

        s[0]=L’C’;s[1]=L’A’;s[2]=L’M’;s[3]=L’W’;s[4]=L’N’;s[5]=L’D’;
                    _ui64tow(user_id,s+6,36);
                    WaitForSingleObject(params->hEvent,INFINITE);
                    SetPropW(MainWindow,s,params->hwnd);
                }
                CloseHandle(params->hEvent);
            }
            if(InterlockedDecrement(&params-
>refcount)==0)HeapFree(Heap,0,params);
        }
}
```