МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДНРАЦИИ

Федеральное государственное бюджетное образовательное

учреждение высшего образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ СИСТЕМ И ТЕХНОЛОГИЙ

Кафедра: «Информационные системы»

Дисциплина: Информационная безопасность

**Лабораторная работа № 1**

**Вариант 15**

Выполнил:

студент группы ПИбд -41

Кочетков Т.С.

Ульяновск, 2022 г.

**Задание**

Разработка программы разграничения полномочий пользователей на основе парольной аутентификации c использованием встроенных криптопровайдеров.

**Содержание задания**

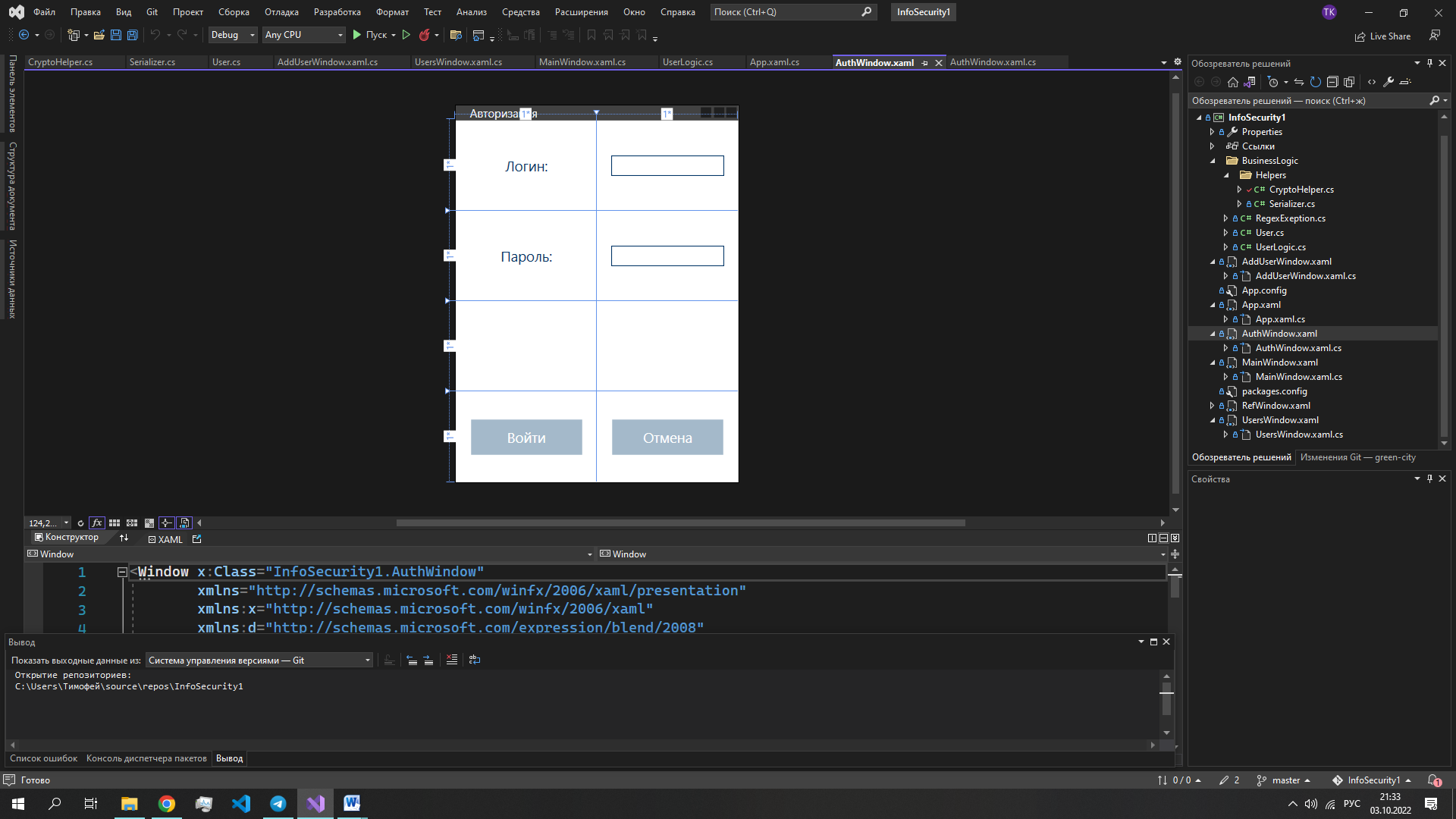
1. Программа должна обеспечивать работу в двух режимах: администратора (пользователя с фиксированным именем ADMIN) и обычного пользователя.
2. В режиме администратора программа должна поддерживать следующие функции (при правильном вводе пароля):
3. смена пароля администратора (при правильном вводе старого пароля);
4. просмотр списка имен зарегистрированных пользователей и установленных для них параметров (блокировка учетной записи, включение ограничений на выбираемые пароли) – всего списка целиком в одном окне или по одному элементу списка с возможностью перемещения к его началу или концу;
5. добавление уникального имени нового пользователя к списку с пустым паролем (строкой нулевой длины);
6. блокирование возможности работы пользователя с заданным именем;
7. включение или отключение ограничений на выбираемые пользователем пароли (в соответствии с индивидуальным заданием, определяемым номером варианта);
8. завершение работы с программой.
9. В режиме обычного пользователя программа должна поддерживать только функции смены пароля пользователя (при правильном вводе старого пароля) и завершения работы, а все остальные функции должны быть заблокированы.
10. После своего запуска программа должна запрашивать у пользователя в специальном окне входа ввод его имени и пароля. При вводе пароля его символы всегда должны на экране заменяться символом ‘\*’.
11. При отсутствии введенного в окне входа имени пользователя в списке зарегистрированных администратором пользователей программа должна выдавать соответствующее сообщение и предоставлять пользователю возможность повторного ввода имени или завершения работы с программой.
12. При неправильном вводе пароля программа должна выдавать соответствующее сообщение и предоставлять пользователю возможность повторного ввода. При трехкратном вводе неверного пароля работа программы должна завершаться.
13. При первоначальном вводе пароля (обязательном при первом входе администратора или пользователя с зарегистрированным ранее администратором именем) и при дальнейшей замене пароля программа должна просить пользователя подтвердить введенный пароль путем его повторного ввода.
14. Если выбранный пользователем пароль не соответствует требуемым ограничениям (при установке соответствующего параметра учетной записи пользователя), то программа должна выдавать соответствующее сообщение и предоставлять пользователю возможность ввода другого пароля, завершения работы с программой (при первом входе данного пользователя) или отказа от смены пароля.
15. Информация о зарегистрированных пользователях, их паролях, отсутствии блокировки их работы с программой, а также включении или отключении ограничений на выбираемые пароли должна сохраняться в специальном файле. При первом запуске программы этот файл должен создаваться автоматически и содержать информацию только об администраторе, имеющем пустой пароль.
16. Файл с учетными данными зарегистрированных пользователей должен быть защищен криптографическими средствами от несанкционированного доступа.
17. При запуске программы файл с учетными данными должен расшифровываться во временный файл (или в файл в оперативной памяти), который после завершения работы программы должен быть снова зашифрован для отражения возможных изменений в учетных записях пользователей. «Старое» содержимое файла учетных записей при этом стирается.
18. После ввода парольной фразы при запуске программы, генерации ключа расшифрования и расшифрования файла с учетными данными зарегистрированных пользователей правильность введенной парольной фразы определяется по наличию в расшифрованном файле учетной записи администратора программы.
19. При вводе неправильной парольной фразы или отказе от ее ввода работа программы должна завершаться с выдачей соответствующего сообщения.
20. Временный файл на диске с расшифрованными учетными данными после завершения работы программы удаляется.
21. Варианты использования алгоритмов шифрования и хеширования выбираются в соответствии с выданным преподавателем заданием.
22. Интерфейс с программой должен быть организован на основе меню, обязательной частью которого должно являться подменю «Справка» с командой «О программе». При выборе этой команды должна выдаваться информация об авторе программы и выданном индивидуальном задании. Интерфейс пользователя программы может также включать панель управления с дублирующими команды меню графическими кнопками и строку состояния.
23. Для реализации указанных в пунктах 2-3 функций в программе должны использоваться специальные диалоговые формы, позволяющие пользователю (администратору) вводить необходимую информацию.

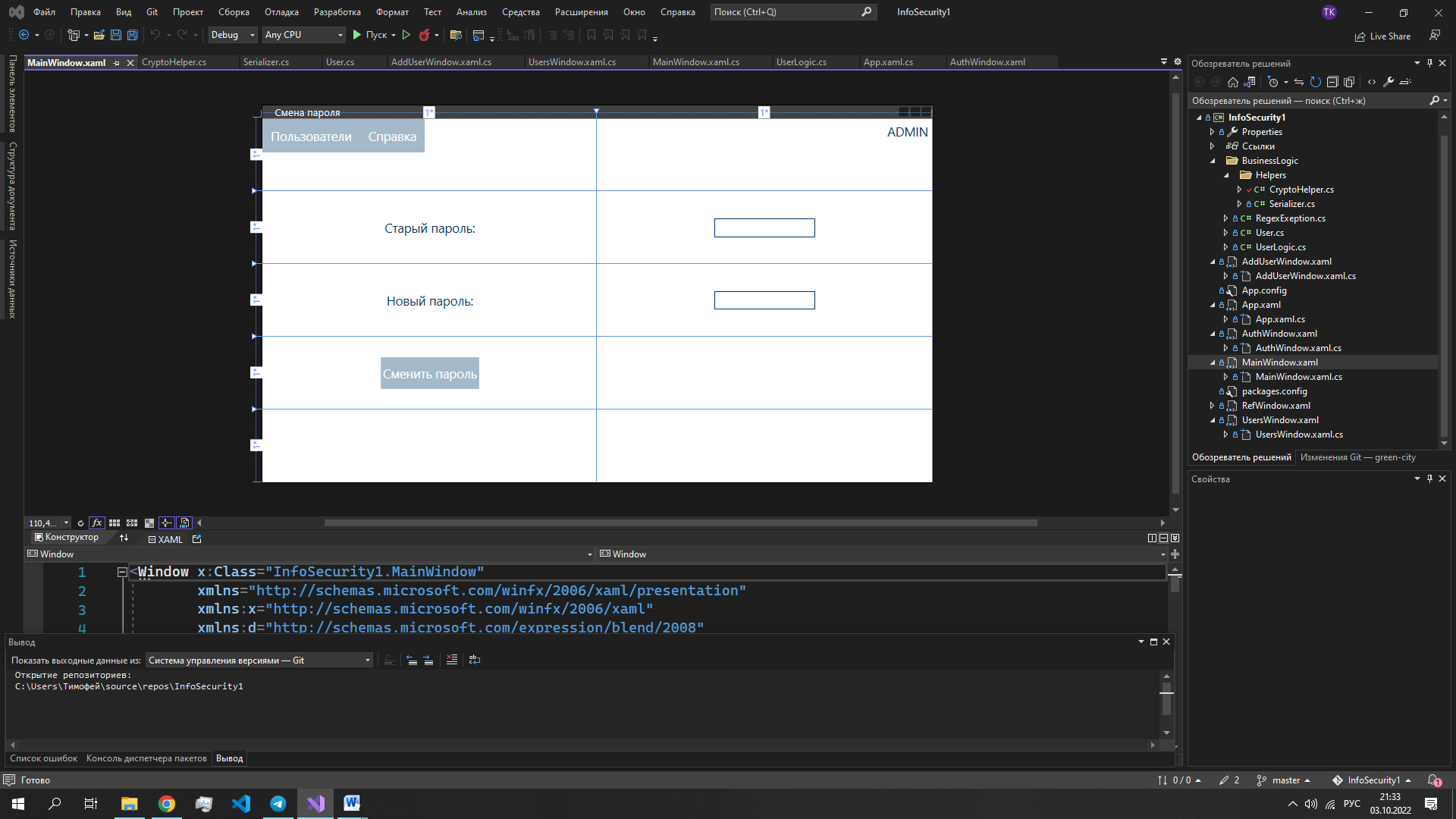
Отчет по лабораторной работе должен содержать листинги программы с подробными комментариями, скрин-шоты экранов, описание структуры файла паролей, подробное описание функций, связанных с шифрованием, описание мест хранения зашифрованных данных, в т.ч. в момент их шифрования(расшифрования)

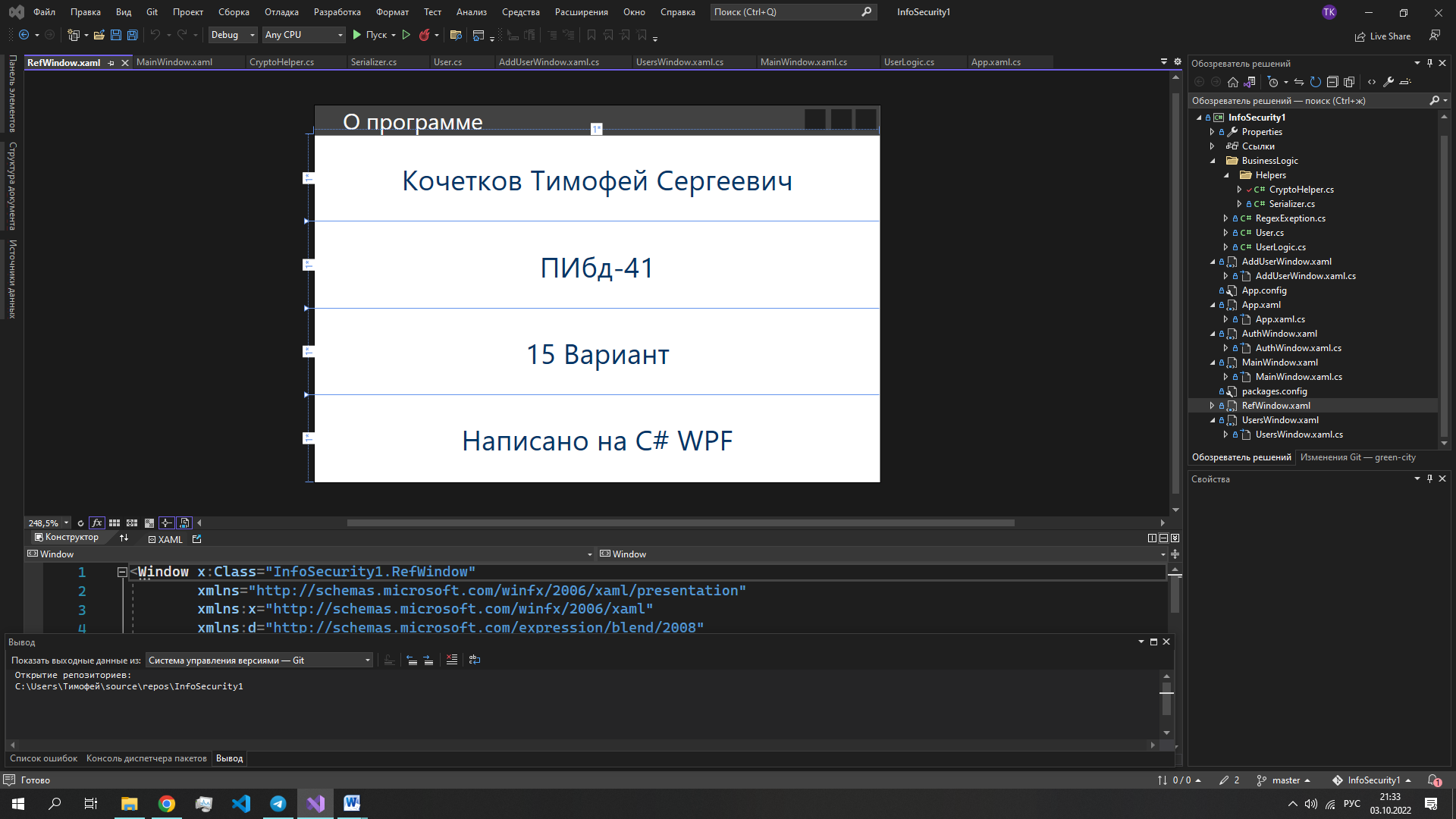
|  |  |  |  |
| --- | --- | --- | --- |
| № | Используемый режим шифрования алгоритма DES | Добавление к ключу случайного значения | Используемый алгоритм хеширования |
|  | 3 | 4 | 5 |
| 15 | CFB | Да | SHA |

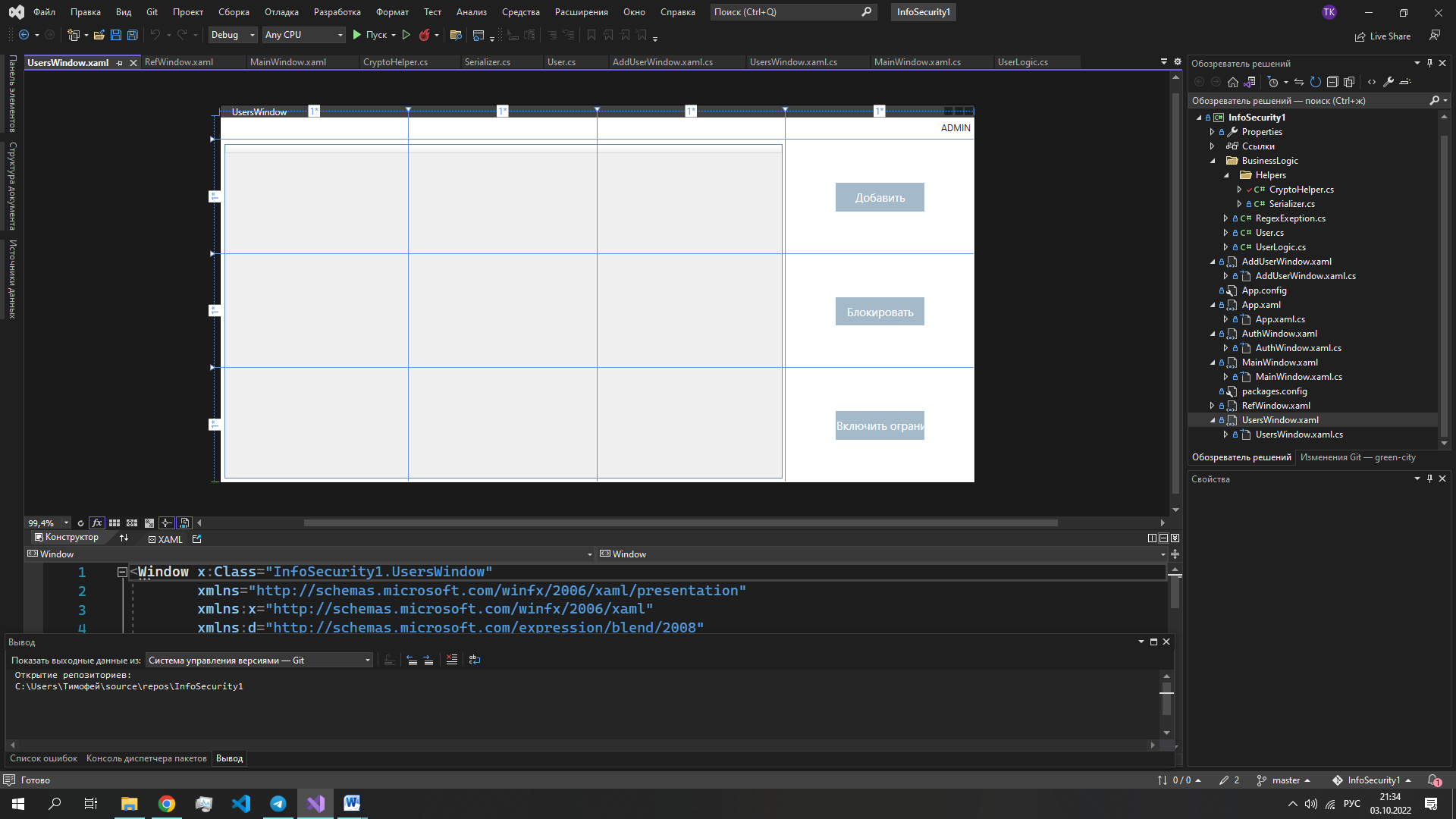
**Реализация**

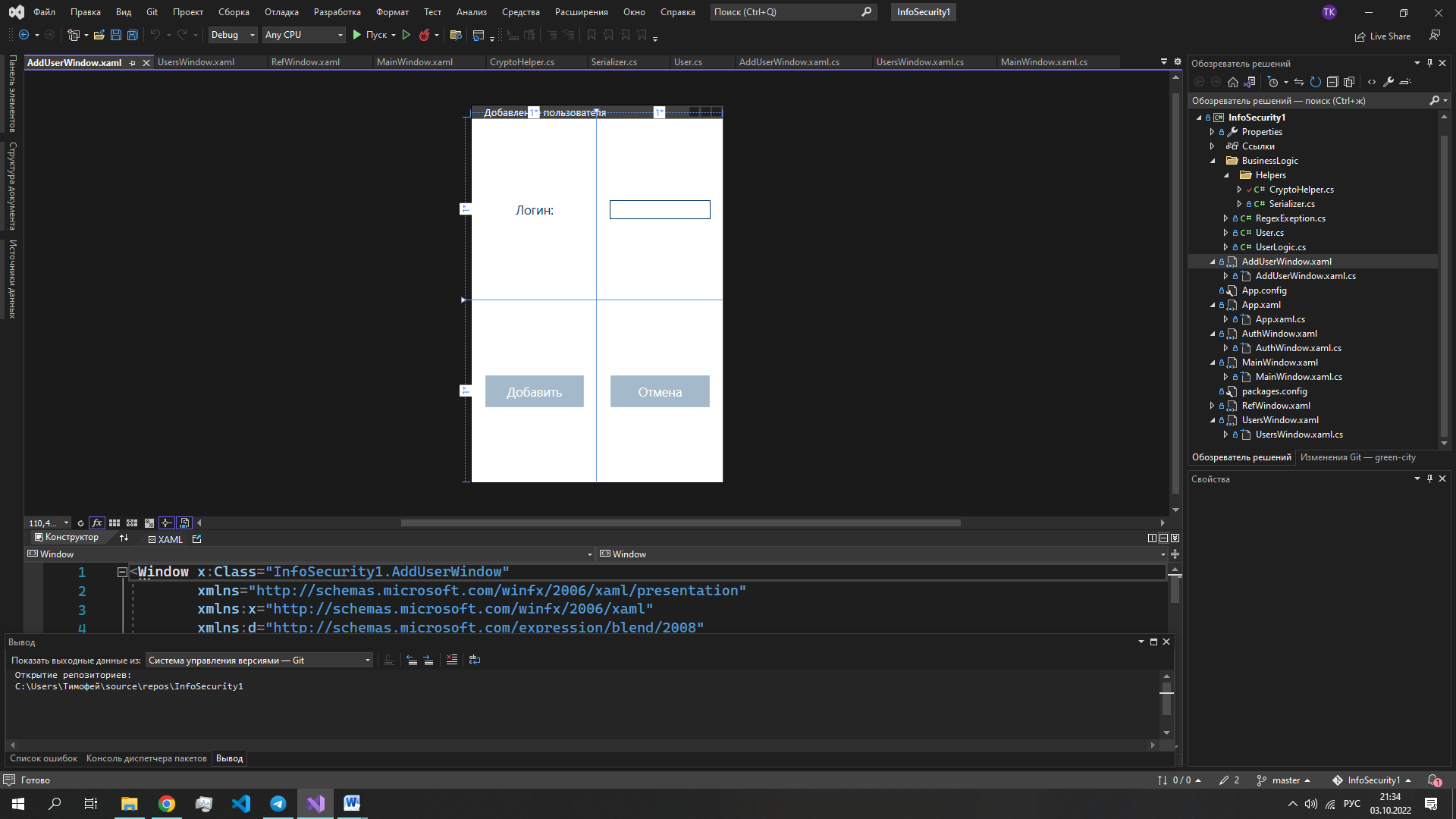
В данной программе созданы несколько интерфейсов пользователей для работы с данными: окно авторизации, основное окно – для смены пароля, окно получения справки, окно для просмотра списка пользователей и его редактирования, окно для добавления пользователя. Вся программа основана на IoC – контейнере Unity (для передачи зависимостей).



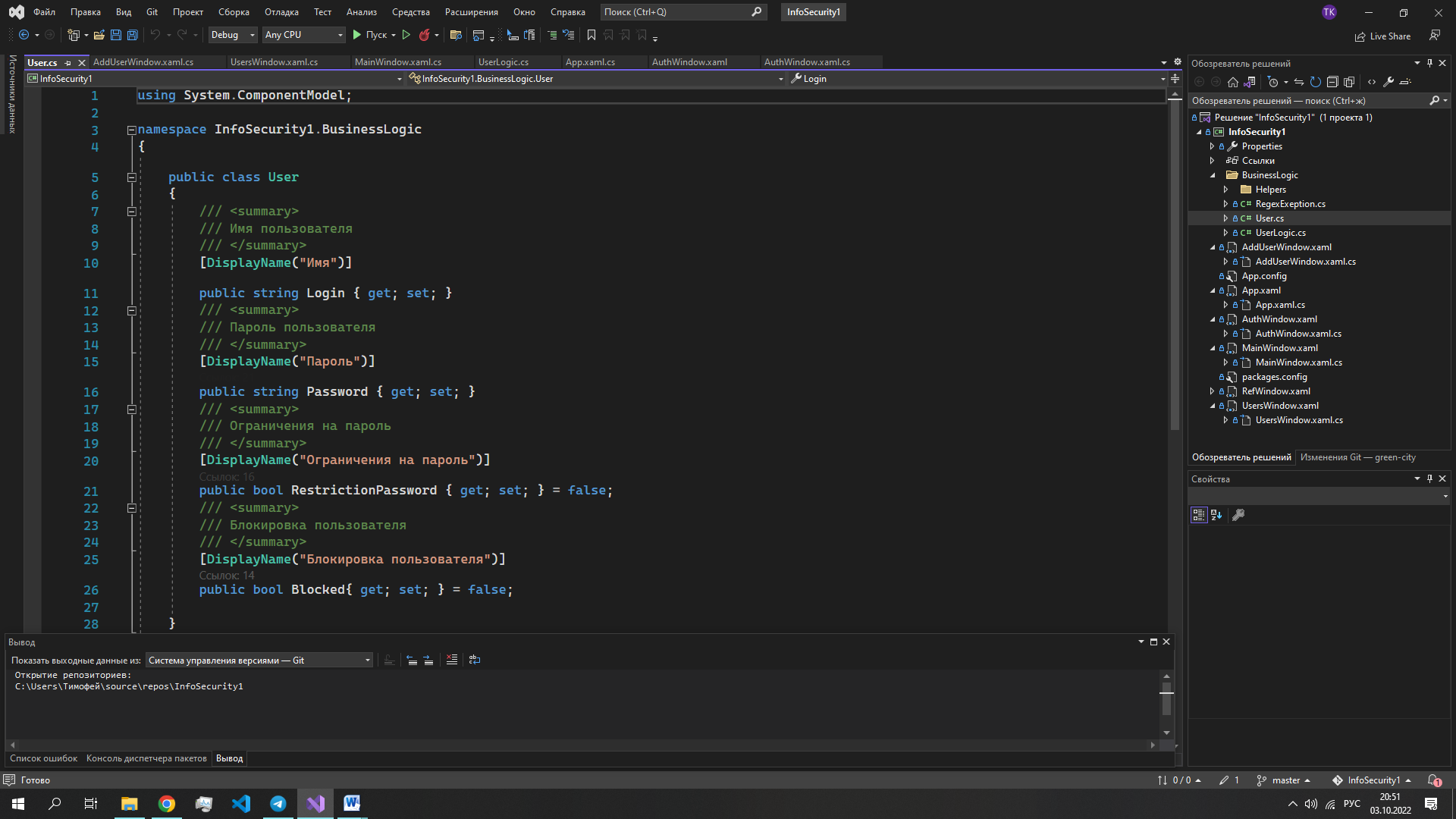






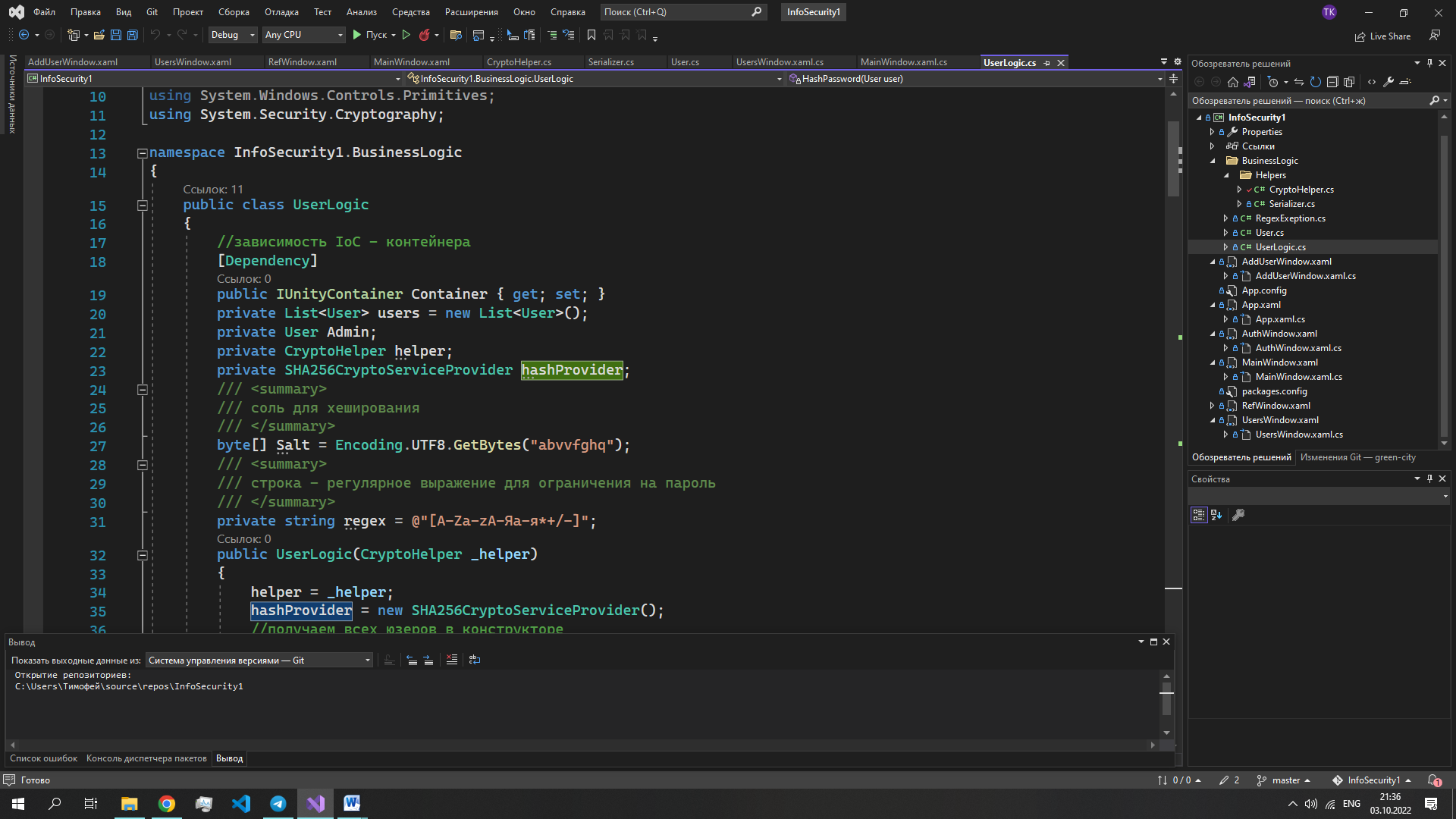


Базовая сущность, которая присваивается каждому пользователю:

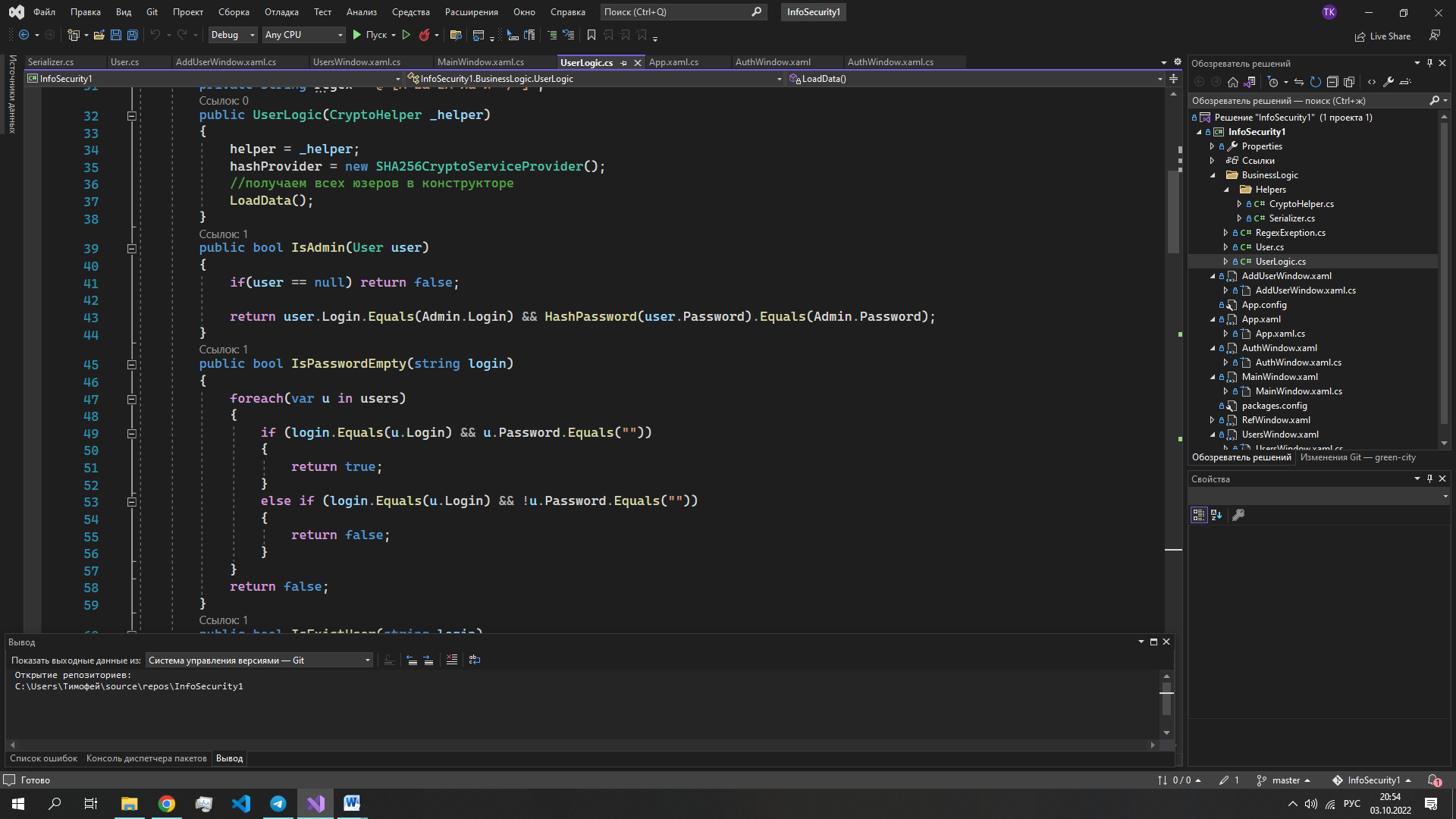


Пользователь имеет 4 свойства – Логин, пароль, ограничения на пароль и блокировка. Атрибуты над свойствами необходимы для корректного отображения списка пользователей в интерфейсе.

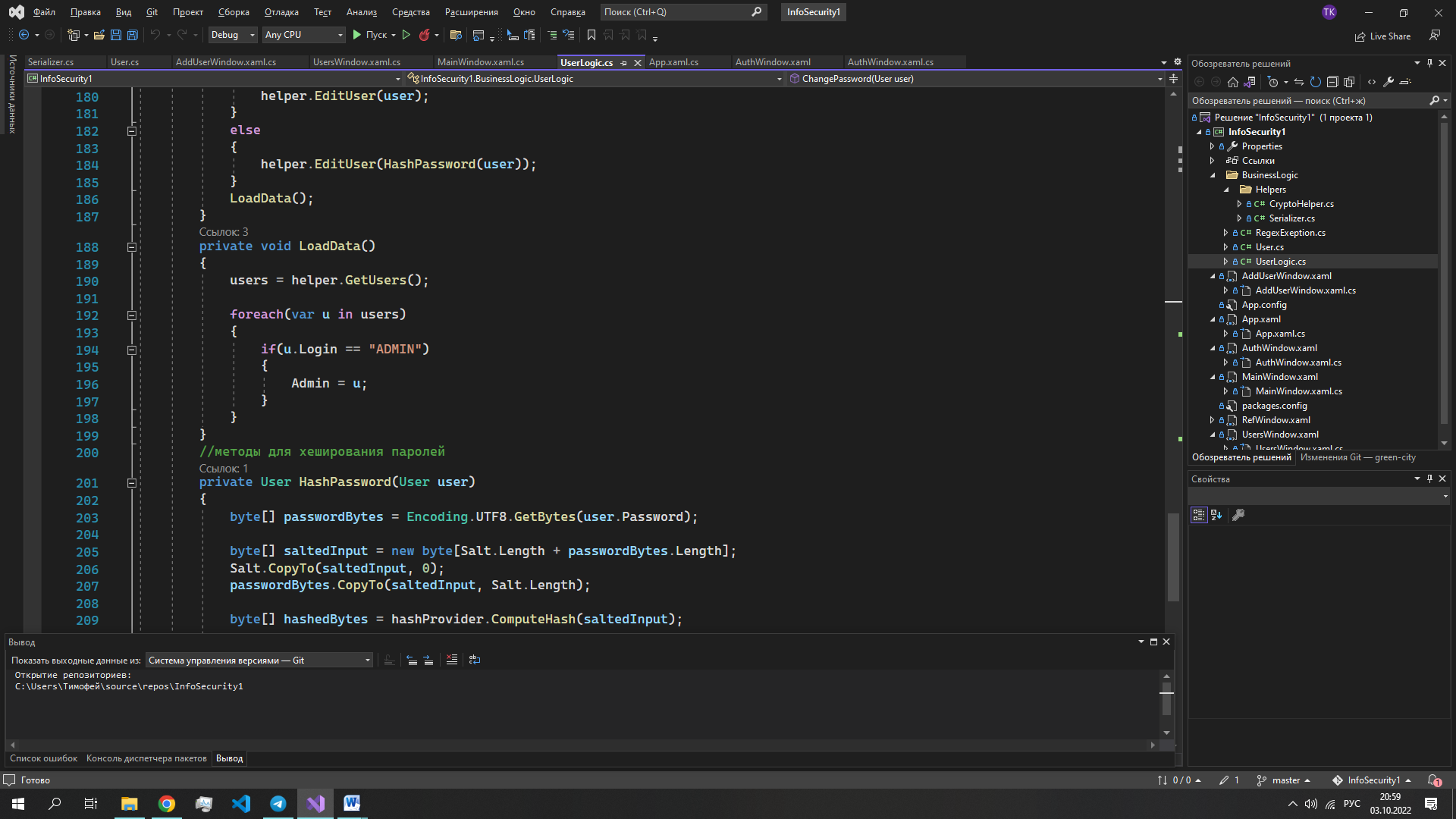
Класс бизнес логики, который взаимодействует со списком пользователей (UsetLogic).



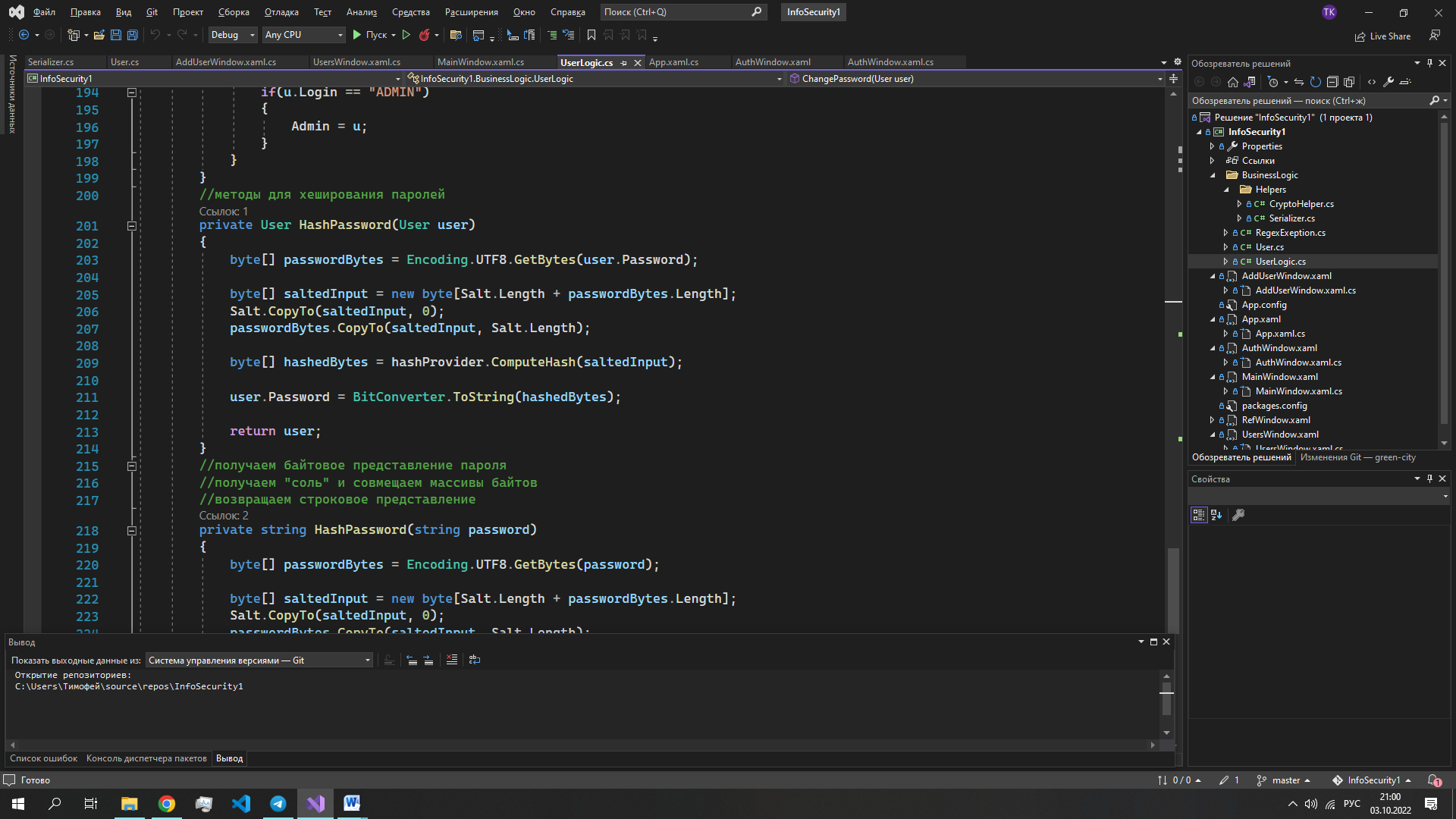
В данном классе есть несколько полей: список пользователей, администратор, класс-шифровщик, класс для хеширования (SHA256CryptoServiceProvider), соль для хеширования, представленная массивом байт, и строка-шаблон для регулярного выражения. Этот шаблон допускает наличие только Латиницы, Кириллицы и знаков арифметических операций в строке (по варианту).

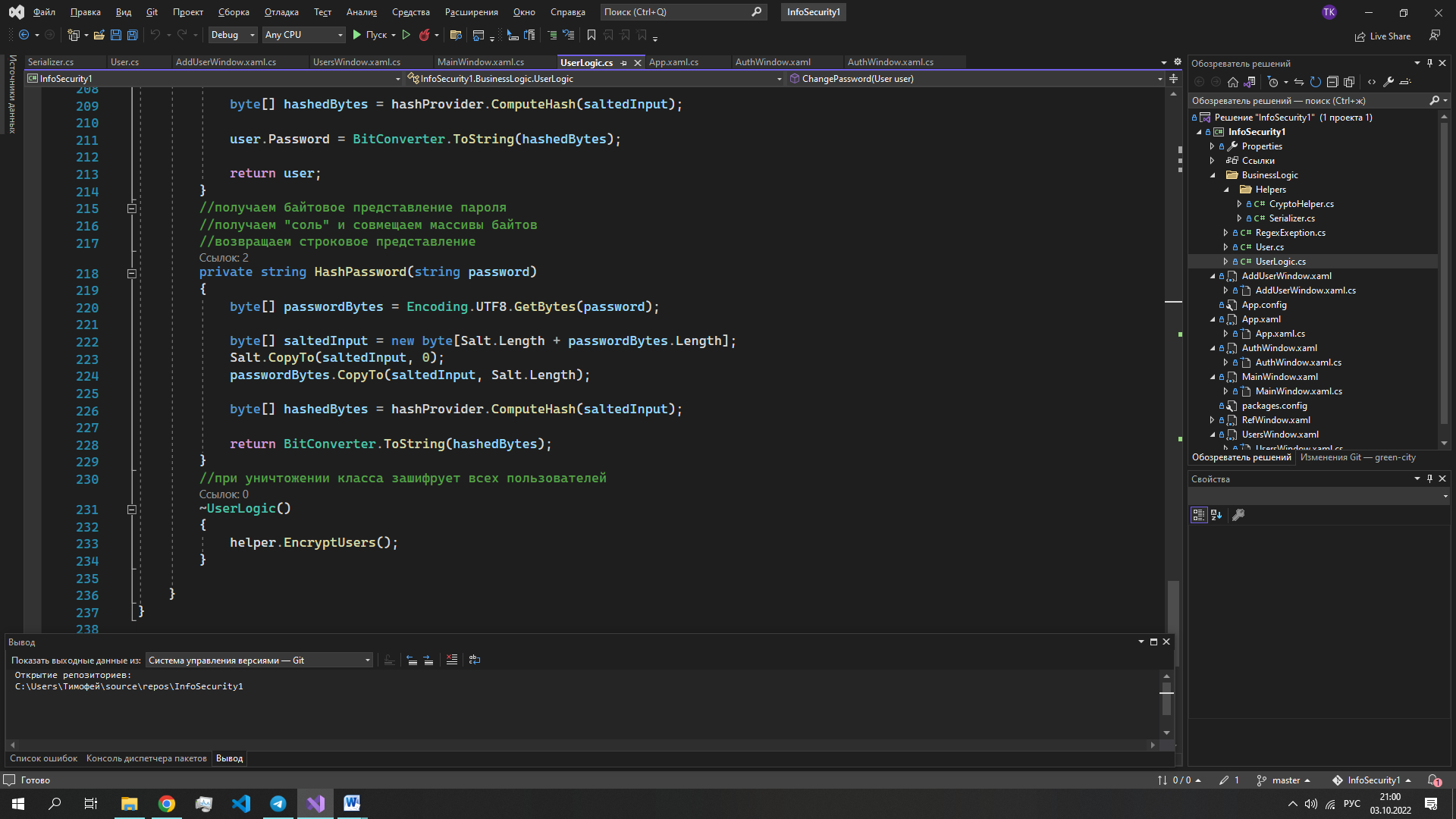


В данном классе реализованы методы проверки на пустоту пароля, проверки эквивалентности пользователя администратору, существования пользователя с таким логином, получение пользователя по логину ( закрытый метод с модификатором private), смена пароля пользователя, метод сравнивания двух паролей, получение списка пользователей, за исключением администратора, редактирование пользователя, а также закрытые методы для загрузки актуального списка пользователей из временного файла – LoadData, метод хеширования паролей пользователя и его перегруженная версия. Класс взаимодействует с классом CryptoHelper – это класс для кодировки списка пользователей и в дальнейшем записи этого списка в файл. Стоит отметить деструктор – в нем с помощью класса-помощника (CryptoHelper) происходит финальная кодировка пользователей. Ниже подробнее описано про этот класс.



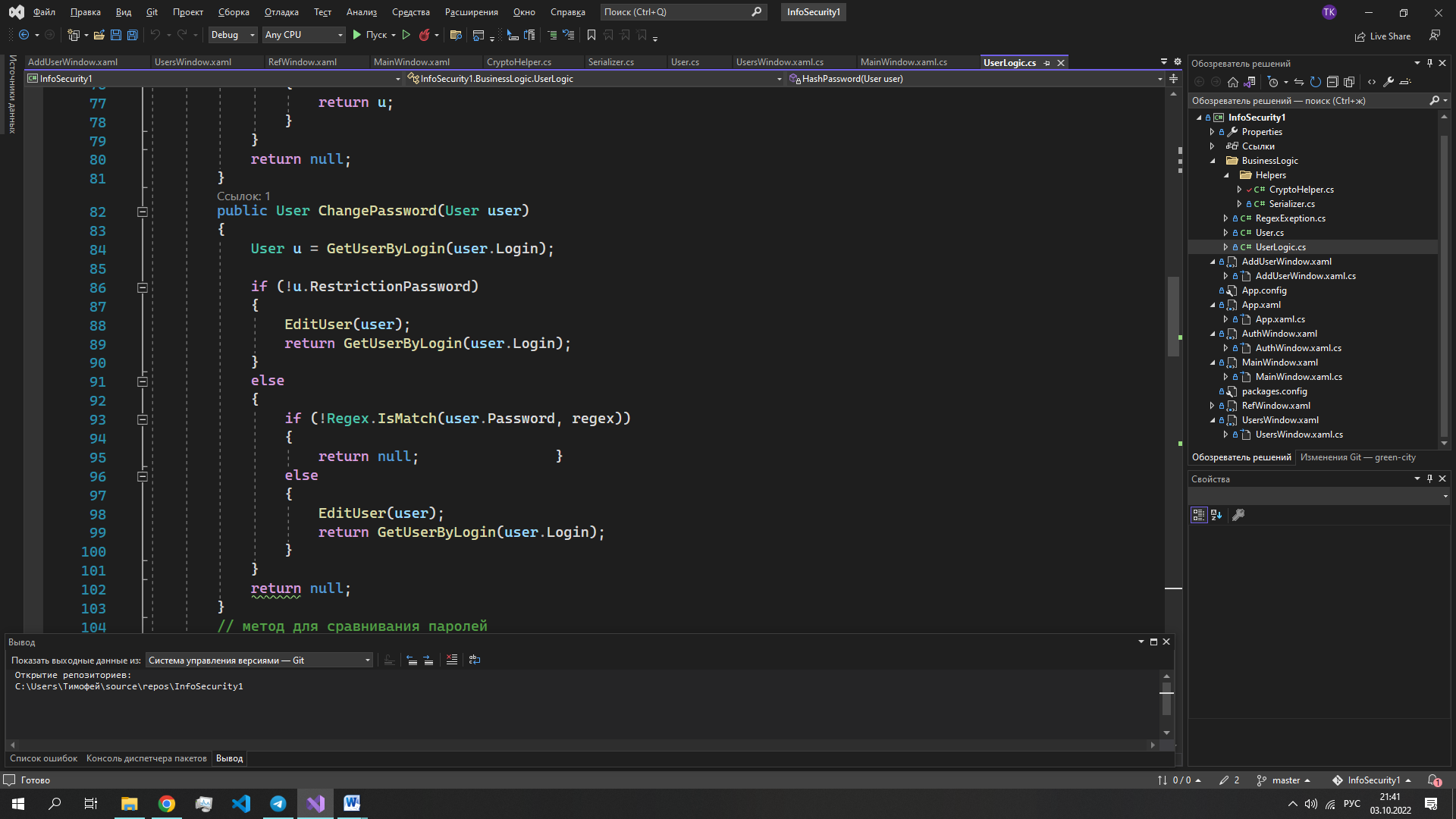
В данном классе реализованы 2 перегруженных метода для хеширования паролей (по пользователю или просто по строке). Стоит отметить, что пароли в программе всегда защищены и передаются в интерфейс администратора в зашифрованном виде.



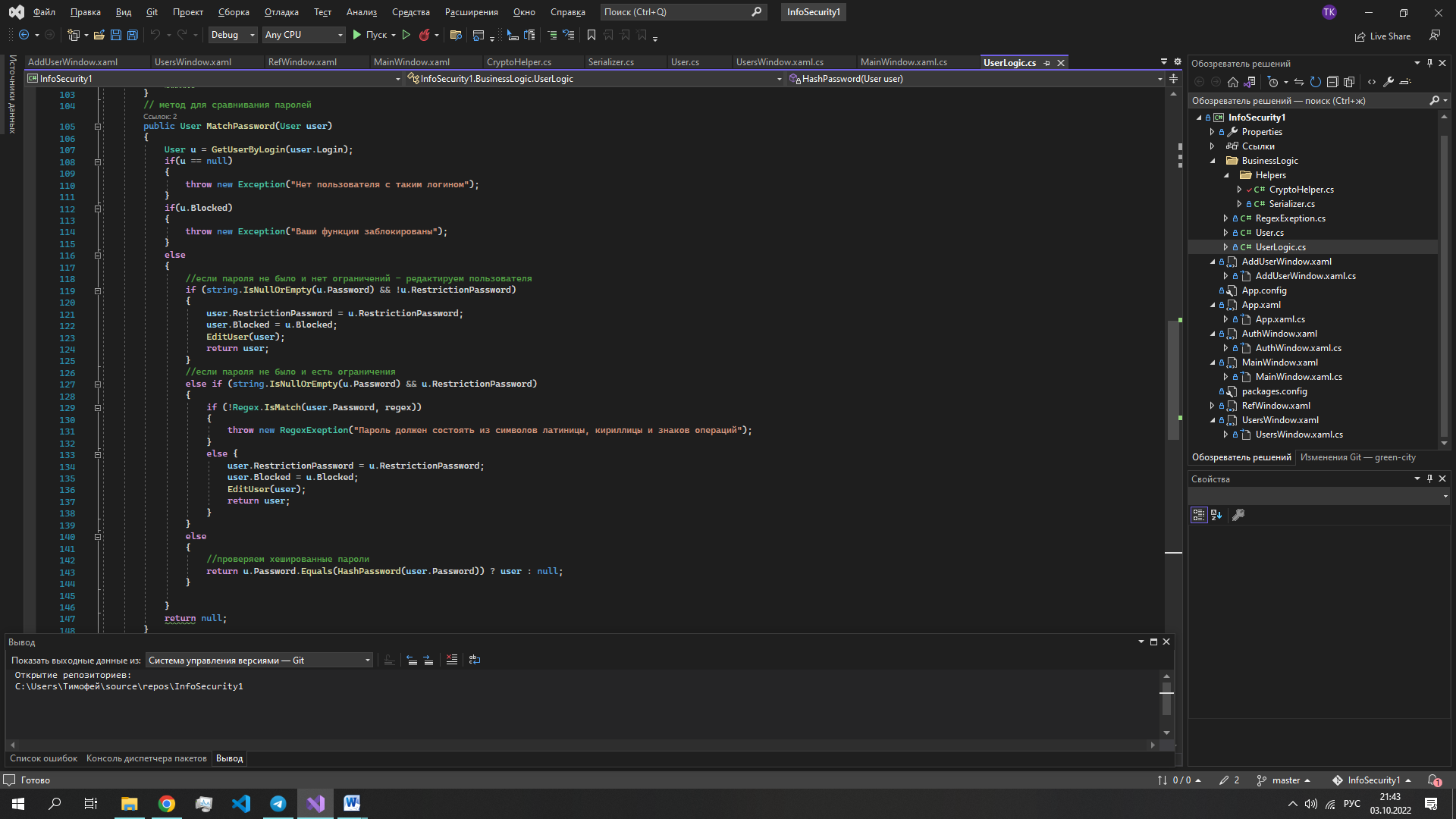


В методе хеширования реализовано добавление “соли” – случайного значения к ключу.

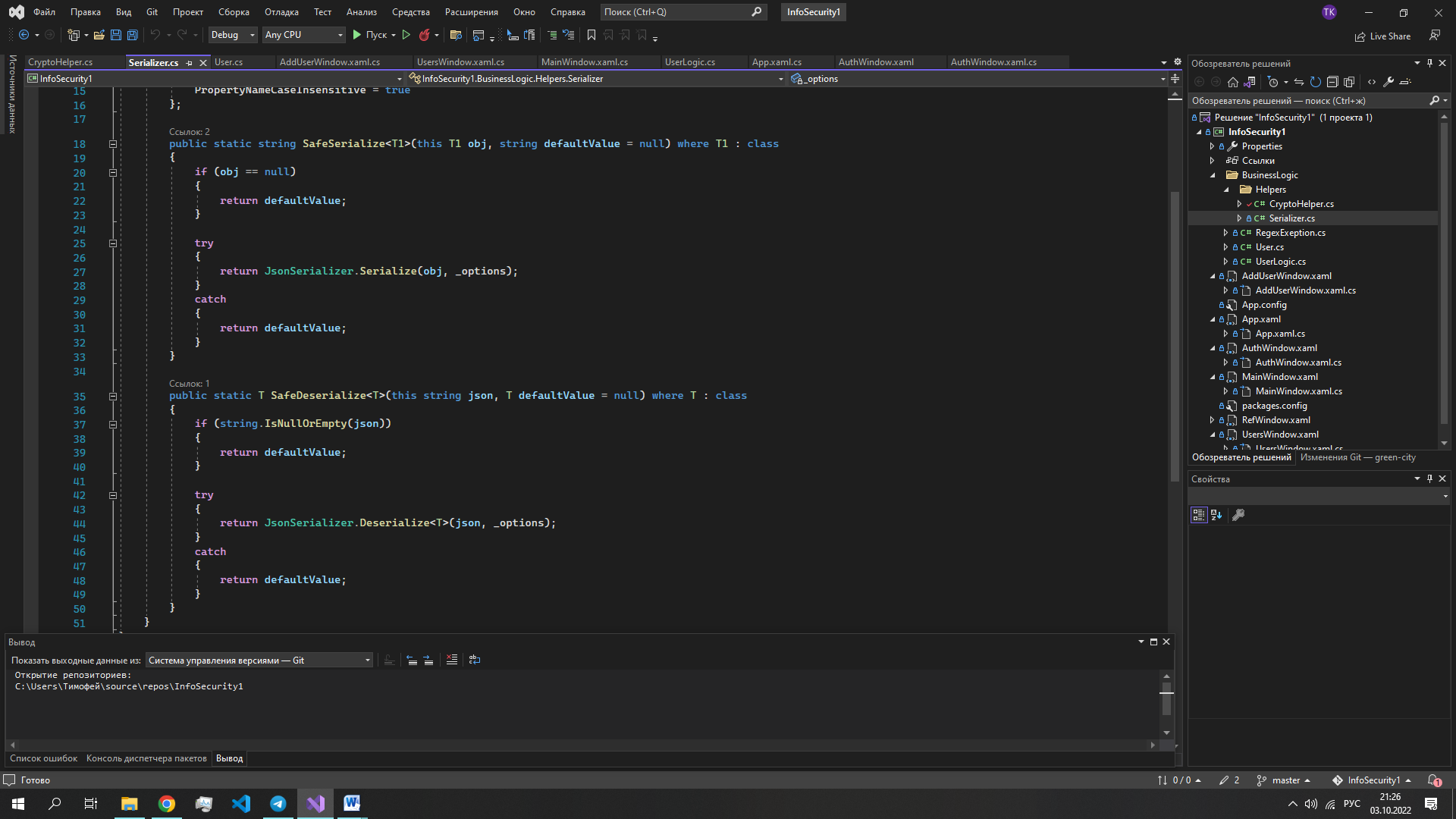
В методе смены пароля, если у пользователя установлено поле RestrictionPassword (ограничение на пароль), тогда происходит проверка по шаблону регулярного выражения.



В методе проверки паролей проверяется наличие пустого пароля у пользователя, и если пароль не был заполнен, в зависимости от ограничений на пароль(есть или нет) происходит процесс валидации пароля по регулярному выражению. Также в этом методу учитывается блокировка пользователя. Данный метод предоставляет пользователю права (авторизация). Стоит отметить, что сравниваются захешированные пароли.

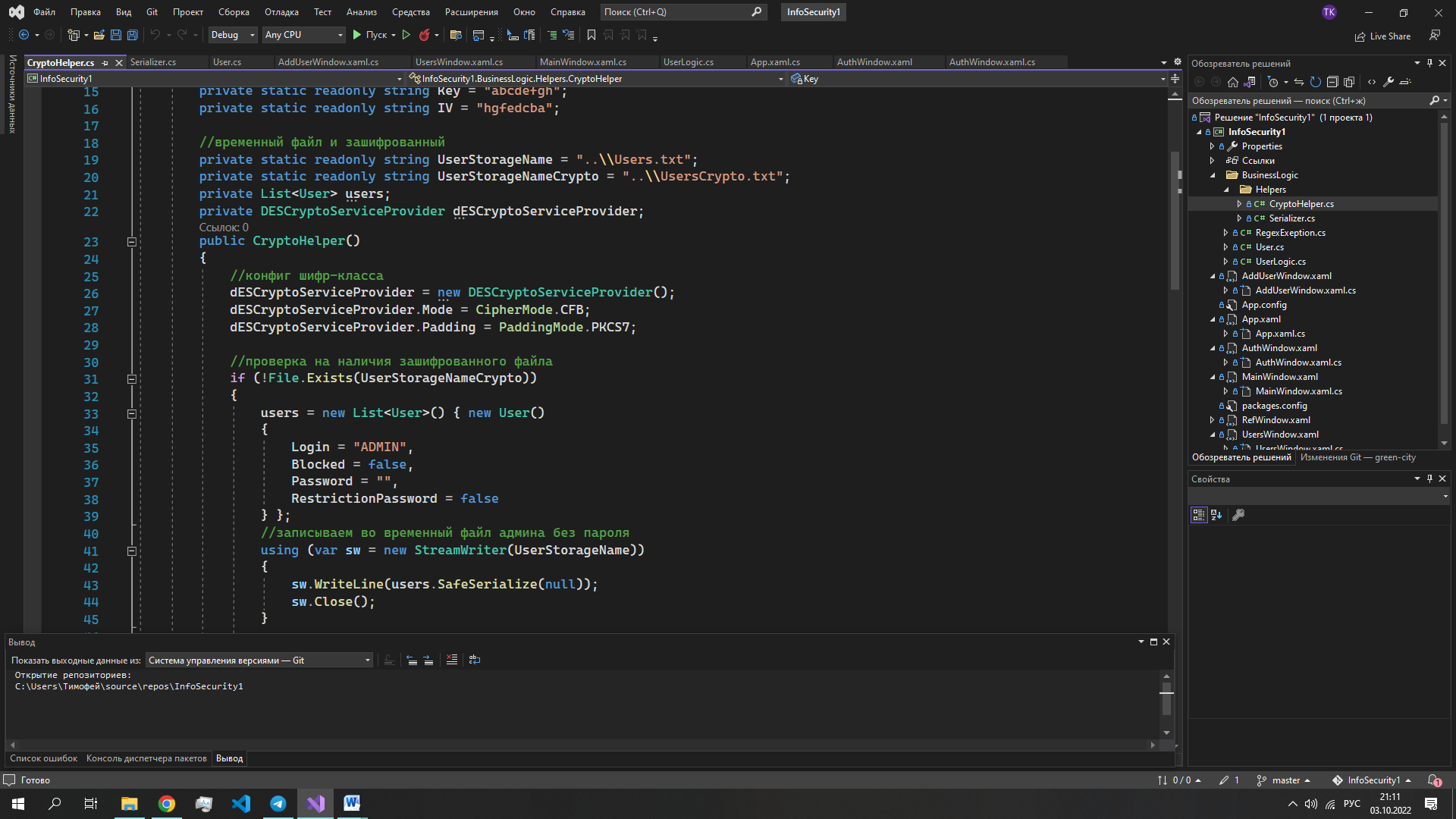


Класс помощник CryptoHelper. Данный класс использует класс-сериализатор, использующий реализацию JSON-сериалайзера (класс Serializer). Статичный класс Serializer имеет 2 статичных обобщенных метода для преобразования объекта в строку и обратно.

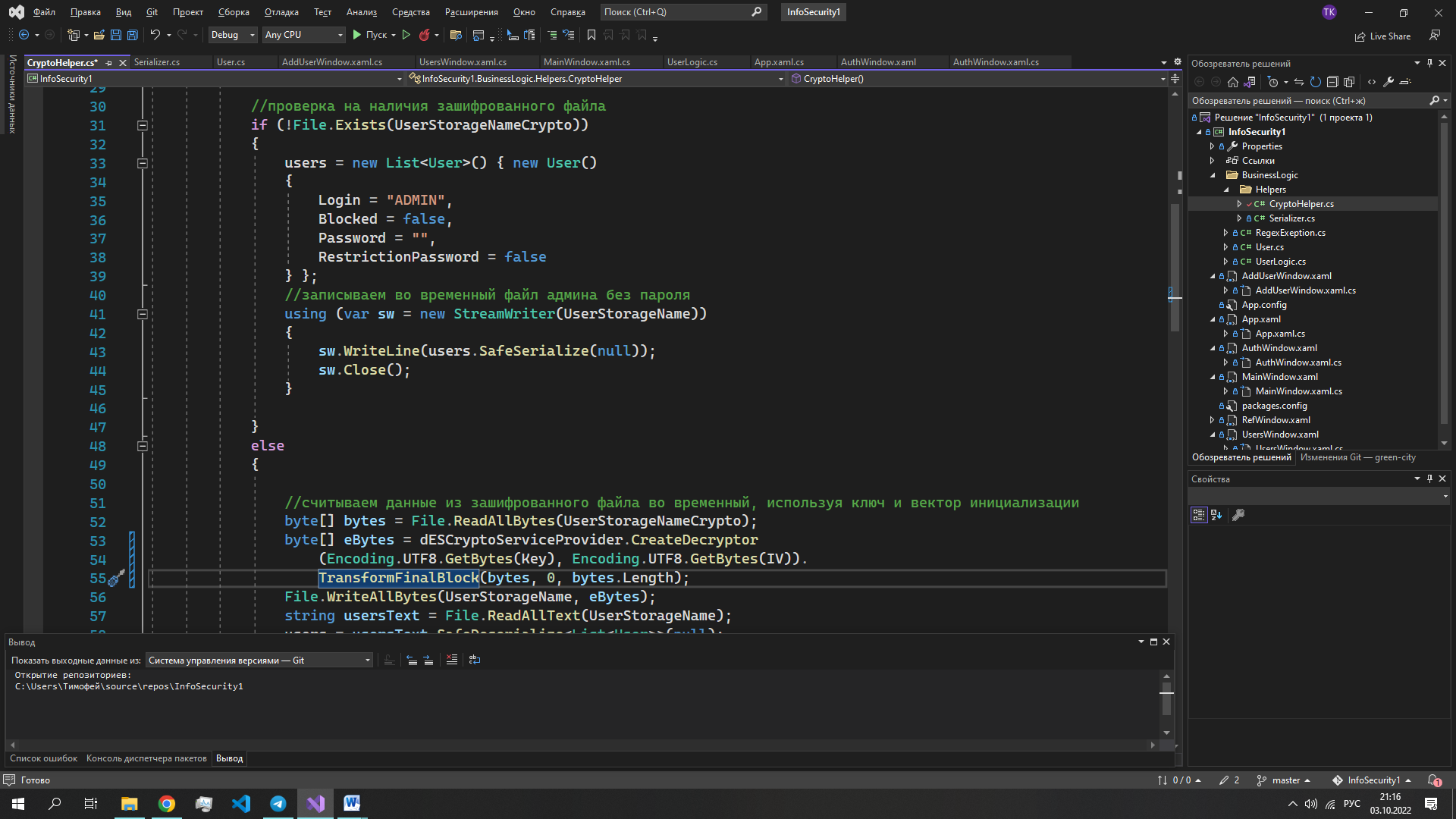




В классе инициализированы поля настроек кодировки (ключ шифрования и вектор инициализации, которые потом будут использованы в методах), пути до файлов(временный и зашифрованный). Также есть класс-кодировщик DesCryptoServiceProvider, обеспечивающий работу с алгоритмом шифрования DES. В констукторе класса настраиваем класс-шифровщик. Используем режим CFB (по заданию) и режим смещения PKCS7, необходимый для тех случаев, когда блок данных короче, чем блок байтов, необходимый для шифрования. В данном режиме добавленный блок состоит из последовательности байтов, каждый из которых равен общему количеству добавленных байтов заполнения.

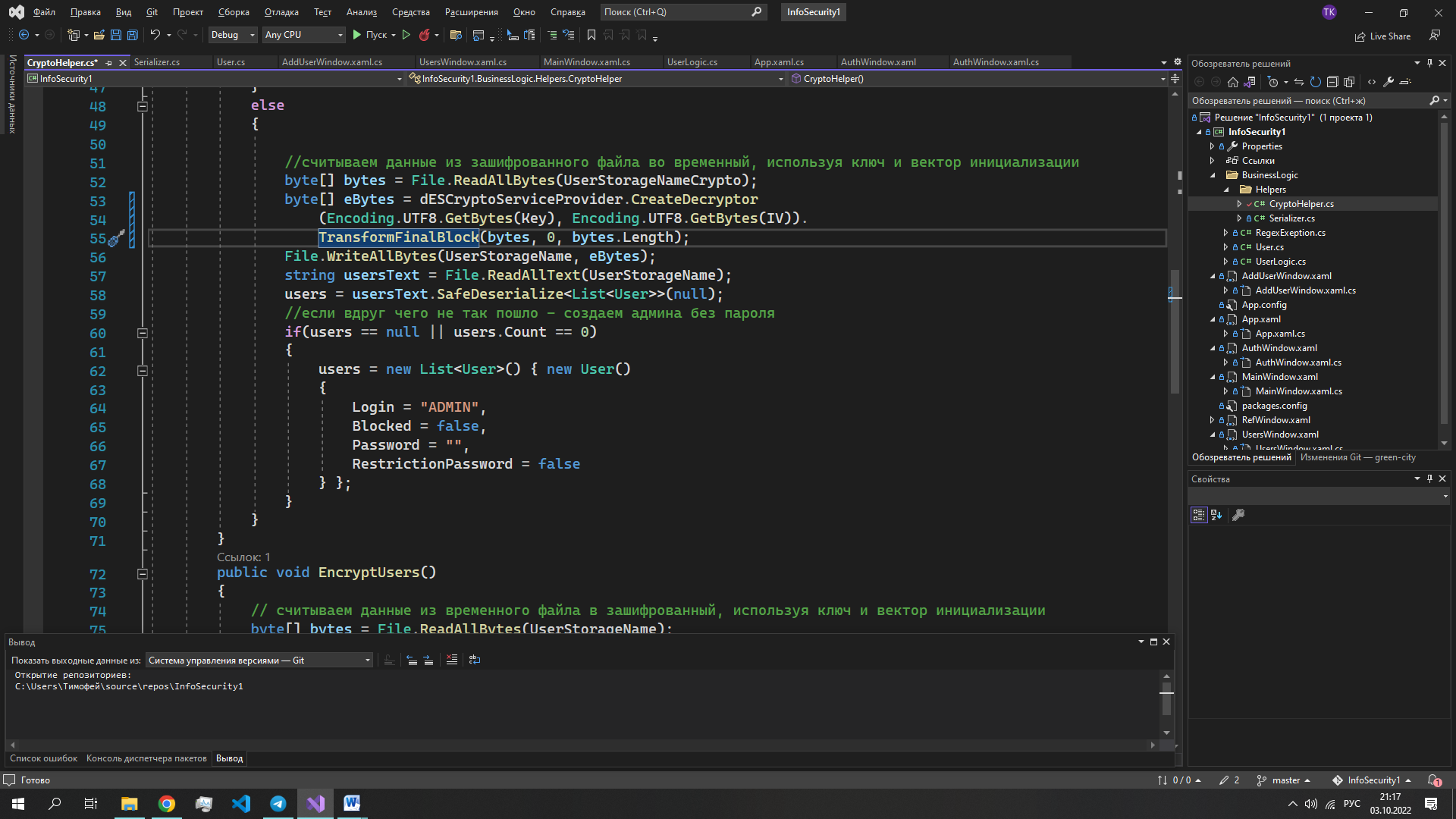


В конструкторе класса (при его создании) идет проверка на наличие зашифрованного файла и создание временного файла. При отсутствии файла создается сущность администратора без пароля.



В случае наличия зашифрованного файла происходит считывание массива байт из файла, дешифровка массива байтов в другой массив, искользуя ключ и вектор инициализации, а затем запись во временный файл.

В случае ошибки (не получилось расшифровать пользователей) создается сущность администратора без пароля. В нормальном поведении программы такого случая не должно быть.



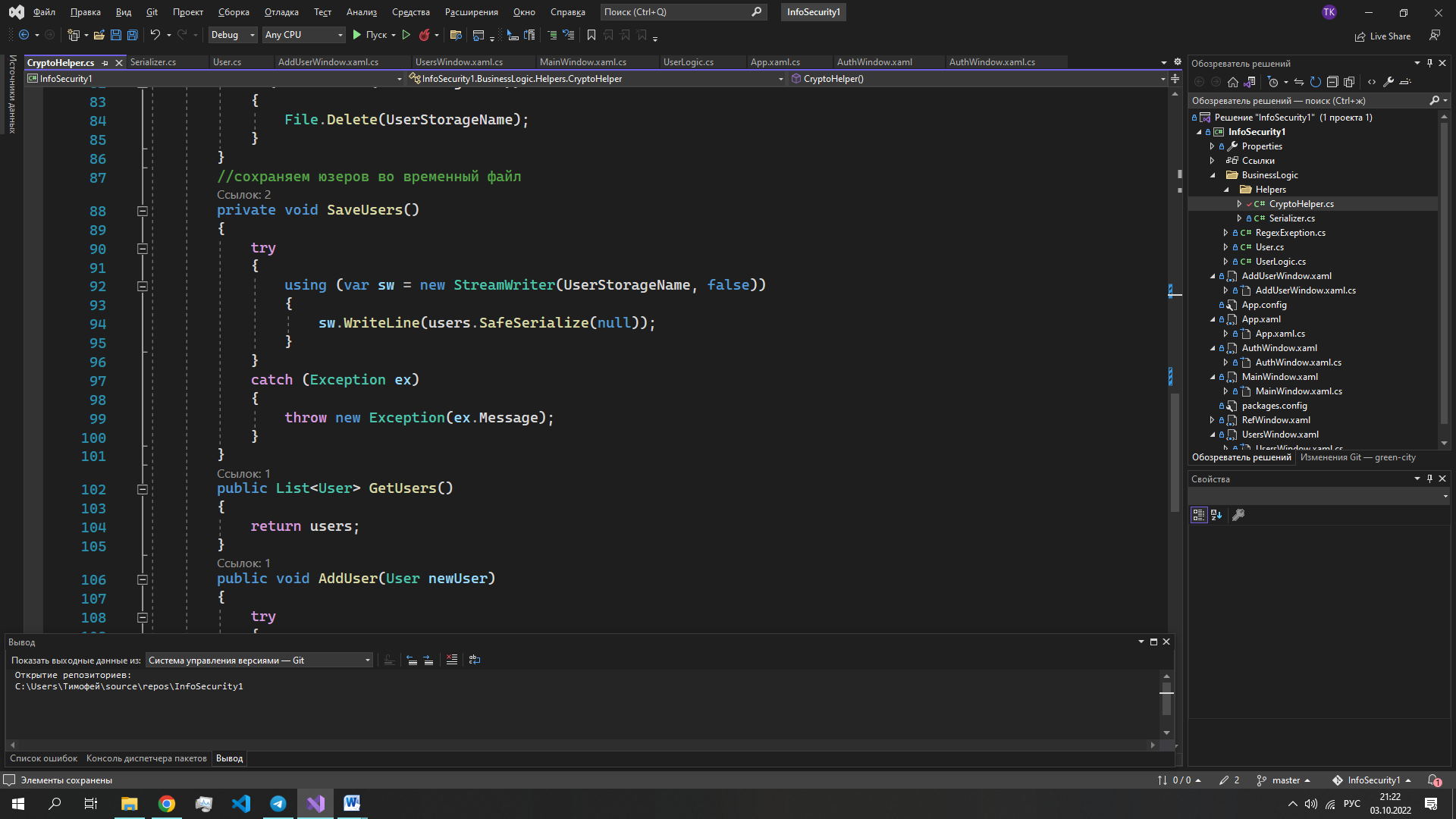
Класс-помощник реализует несколько методов:

Метод кодировки всех пользователей:



В данном методе считывается массив байтов из временного файла, шифруется данный массив, с помощью ключа и вектора инициализации, затем записывается в финальный файл, затем удаляется временный файл.

Методы получения списка пользователей и сохранение списка во временный файл:



А также методы для добавления и редактирования пользователя:

