Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

Кафедра БиУТ

Допустить к защите зав. кафедрой

_____ /С.Н. Новиков /

# ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА СПЕЦИАЛИСТА

Разработка системы дистанционного электронного голосования

Пояснительная записка

Студент _____ / А.А. Крылосов /

Институт ИВТ Группа АБ-66

Руководитель _____ / Г.В. Попков /

Консультанты:

– по экономическому обоснованию

_____/_____/

– по безопасности жизнедеятельности

_____/ _____/

Рецензент_____ / _____/

Новосибирск 2022

Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

**КАФЕДРА**

Безопасность и управление в телекоммуникациях

**ЗАДАНИЕ**

**НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ СПЕЦИАЛИСТА**

СТУДЕНТА А.А. Крылосова ГРУППЫ АБ-66

«УТВЕРЖДАЮ»

« 24 » мая 2021 г.

Зав. кафедрой БиУТ

_____/ С.Н. Новиков /

Новосибирск 2021

1. Тема выпускной квалификационной работы специалиста:

Разработка системы дистанционного электронного голосования

утверждена приказом по университету от « 24 » мая 2021 г. № 4/823о-21

2. Срок сдачи студентом законченной работы « 19 » января 2022 г.

3. Исходные данные по проекту (эксплуатационно-технические данные, техническое задание):

Язык программирования Python 3 и его документация

Python библиотеки: Flask, Tkinter

Облачная PaaS-платформа Heroku

База данных Postgresql

| 4. Содержание расчетно-пояснительной записки (перечень подлежащих разработке вопросов) | Сроки выполнения по разделам |
|---|---|
| Введение | 13.09.2021 г. |
| 1. Анализ предметной области | 11.10.2021 г. |
| 2. Разработка технического задания | 08.11.2021 г. |
| 3. Разработка системы дистанционного электронного голосования | 06.12.2021 г. |
| 4. Безопасность жизнедеятельности | 13.12.2021 г. |
| 5. Технико-экономическое обоснование работы | 20.12.2021 г. |
| 6. Заключение | 27.12.2021 г. |
| 7. Список литературы | 09.01.2022 г. |
| 8. Приложения | 15.01.2022 г. |

Консультанты по ВКР (с указанием относящихся к ним разделов):

1. Раздел по технико-экономическому обоснованию

_____

_____

_____

_____

_____

_____

_____/_____/

2. Раздел по безопасности жизнедеятельности

_____

_____

_____

_____

_____

_____

_____

_____/_____/

| Дата выдачи задания | Задание принял к исполнению |
| --- | --- |
| « 01 » сентября 2021 г. | « 01 » сентября 2021 г. |
| _____/ Г.В. Попков / | _____/ А.А. Крылосов / |
| (подпись, Ф.И.О. руководителя) | (подпись, Ф.И.О. студента) |

Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

---

# РЕЦЕНЗИЯ

на выпускную квалификационную работу студента А.А. Крылосова
по теме «Разработка системы дистанционного электронного голосования»
В настоящее время наблюдается тенденция к переводу многих привычных процессов на дистанционную основу, что касается в том числе и процесса голосования. Современные протоколы для организации дистанционного голосования позволяют решить большинство проблем присущих обычному голосованию и обеспечить гарантированную и математически обоснованную защищённость. Поэтому считаю, что работа Крылосова А.А. является актуальной.
Разработанная система электронного голосования является работоспособной и может применяться для проведения тайного голосования. Система является защищенной ко многим уязвимостям, предоставлено подробное описание протокола голосования и алгоритмов работы системы.
В качестве замечания можно отметить, что в рамках ВКР не было проведено массового тестирования в условиях реальной работы с большим числом пользователей, что не позволяет называть систему готовой к реальному применению. Кроме того, в тексте пояснительной записки присутствуют незначительные грамматические и стилистические ошибки.
Тем не менее, несмотря на замечания, считаю, что работа выполнена на высоком уровне, студент справился с поставленной задачей и заслуживает оценки «отлично» и присвоения квалификации специалист по защите информации по

специальности 10.05.02 «Информационная безопасность телекоммуникационных систем».

Доц. каф. ПМиК, к.т.н. Ракитский Антон Андреевич

« 13 » января 2022 г.

С Рецензией ознакомлен _____ /А.А. Крылосов/

« 13 » января 2022 г.

Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации
федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

---

**ОТЗЫВ**

О работе студента А.А. Крылосова в период подготовки выпускной квалификационной работы по теме «Разработка системы дистанционного электронного голосования»

| | | | |
|---|---|---|---|
| Работа имеет практическую ценность | | Тема предложена предприятием | |
| Работа внедрена | | Тема предложена студентом | |
| Рекомендую работу к внедрению | | Тема является фундаментальной | |
| Рекомендую работу к опубликованию | | Рекомендую студента в магистратуру | |
| Работа выполнена с применением ЭВМ | | Рекомендую студента в аспирантуру | |

Руководитель выпускной квалификационной работы специалиста
Доц. каф. БиУТ, к.т.н. Глеб Владимирович Попков
« 15 » января 2022 г.
С Отзывом ознакомлен /А.А. Крылосов/
« 15 » января 2022 г.

Приложение к Отзыву

**Уровень сформированности компетенций у студента**

А.А. Крылосова

| Компетенции | | Уровень сформированности компетенций | | |
|---|---|---|---|---|
| | | высокий | средний | низкий |
| 1 | | 2 | 3 | 4 |
| Профессиональные | ПК-1 - способностью осуществлять анализ научно-технической информации, нормативных и методических материалов по методам обеспечения информационной безопасности телекоммуникационных систем | | | |
| | ПК-5 - способностью проектировать защищенные телекоммуникационные системы и их элементы, проводить анализ проектных решений по обеспечению заданного уровня безопасности и требуемого качества обслуживания, разрабатывать необходимую техническую документацию с учетом действующих нормативных и методических документов | | | |
| | ПК-7 - способностью осуществлять рациональный выбор средств обеспечения информационной безопасности телекоммуникационных систем с учетом предъявляемых к ним требований качества обслуживания и качества функционирования | | | |
| | ПК-12 - способностью выполнять технико-экономические обоснования, оценивать затраты и результаты деятельности организации в области обеспечения информационной безопасности | | | |

# АННОТАЦИЯ

Выпускной квалификационной работа студента А.А. Крылосова

по теме Разработка системы дистанционного электронного голосования

Объём работы – 90 страниц, на которых размещены 16 рисунков и 12 таблиц. При написании работы использовалось 9 источников.

Ключевые слова: электронное голосование, система защиты информации, персональные данные, аутентификация, базы данных, протоколы голосования.

Работа выполнена на: кафедре БиУТ СибГУТИ

Руководитель: доц. каф. БиУТ Попков Г.В.

Целью работы разработка системы дистанционного электронного голосования

Решаемые задачи: анализ предметной области, разработка технического задания, разработка системы дистанционного электронного голосования, безопасность жизнедеятельности, технико-экономическое обоснование работы.

Основные результаты: система дистанционного электронного голосования

**Graduation thesis abstract**

of A.A. Krylosov on the theme Development of a remote electronic voting system

The paper consists of 90 pages, with 16 figures and 12 tables/charts/diagrams. While writing the thesis 9 reference sources were used.

Keywords: electronic voting, information security system, personal data, authentication, databases, voting protocols.

The thesis was written at BIUT department SibSUTIS

(name of organization or department)

Scientific supervisor associate professor of the BiUT Popkov G.V.

The goal/subject of the paper is Development of a remote electronic voting system

Tasks: analysis of the subject area, development of technical specifications, development of a remote electronic voting system, life safety, feasibility study of work

Results remote electronic voting system

ОГЛАВЛЕНИЕ



Подп. и дата | Инв. № дубл. | Взам. инв. № | Подп. и дата | Инв. № подл

| Из | Лист | № докум. | Подп. | Дата |
|---|---|---|---|---|
| Разраб. | | *А.А. Крылосов* | | |
| Пров. | | *Г.В. Попков* | | |
| Н/контр | | | | |
| Рецензент | | *А.А. Ракитиский* | | |
| Утвердил | | *С.В. Новиков* | | |

*ИИВТ.10.05.02.066 ПЗ*

Разработка системы дистанционного электронного голосования

Содержание

| Лит | Лист | Листов |
|---|---|---|
| | 2 | 90 |

Введение

Развитие институтов выборов свидетельствует о тенденции перехода мировых стран на электронное голосование. В настоящее время системы электронного голосования становятся востребованными во многих странах мира, в ряде стран – они уже внедрены в избирательную практику. Свой опыт имеют США, Великобритания, Индия, Нидерланды, Бразилия, Бельгия, Венесуэла, Португалия, Испания, Филиппины, Эстония, Швейцария, Австрия, Австралия, Норвегия, Япония.

Результатом исследований и экспериментов стали выводы о неоспоримых преимуществах электронного голосования:

- значительное ускорение подведения итогов голосования;

- отсутствие ошибок при подсчете бюллетеней;

- обеспечение принципа «прозрачности» выборов;

- облегчение труда избирательных комиссий, снижение рисков от ошибок, связанных с усталостью;

- экономия бумаги и возможность оперативного изменения списков без перепечатывания всего тиража бюллетеней;

- использование многоязычных интерфейсов. [1]

Однако, при этом возникает ряд специфических проблем, препятствующих честности выборов. Например, сомнения в истинности результатов, полученных с помощью машин. Также дистанционно намного сложнее авторизовать избирателя или удостовериться, что на ход голосования никто не повлиял.

Целью данной выпускной квалификационной работы является разработка системы дистанционного электронного голосования, которая бы отвечала необходимым требованиям и позволяла проводить прозрачные и честные выборы.

Для этого необходимо решить следующие задачи:

- определить объект разработки, составить модель угроз и нарушителя;
- разработать техническое решение, выбрать протокол голосования;
- написать исходный код системы электронного голосования;

– рассмотреть вопросы безопасности жизнедеятельности;

– выполнить технико-экономические расчеты.

# 1 Анализ предметной области

## 1.1 Постановка задачи

В данной главе необходимо определить объект разработки и описать его возможности. Произвести сравнительный анализ существующих систем голосования. Разработать модели потенциальных угроз и нарушителя, на основе которых будет строиться система защиты разрабатываемого веб-приложения.

## 1.2 Определение объекта разработки

Понятие «электронное голосование» можно определить как набор различных способов волеизъявления избирателя, объединенных одним обязательным условием: подсчет голосов производится при помощи специальных программно-технических устройств без вмешательства человека. В постановлении ЦИК России от 27 августа 2014 года № 248/1529–6 «О Порядке электронного голосования с использованием комплексов для электронного голосования на выборах, проводимых в Российской Федерации» в разделе 1.1 дается определение этому понятию: «Электронное голосование – голосование без использования бюллетеня, изготовленного на бумажном носителе, с использованием комплекса средств автоматизации ГАС «Выборы». Таким образом, в России электронным голосованием не является голосование с использованием оптических машин сканирования бумажных бюллетеней.

Электронное голосование часто рассматривается как инструмент повышения эффективности избирательного процесса и повышения доверия к нему. Правильно реализованные решения для электронного голосования могут повысить безопасность бюллетеня, ускорить обработку результатов и упростить само голосование.

Как и с бумажным голосованием система для дистанционного электронного голосования должна обеспечить:

– голосование только легитимных участников и при том, только один раз;

– тайну голосования, никто, кроме голосующего, не должен знать его выбор;

– аудит списка избирателей (поимённый перечень проголосовавших);

– аудит результатов голосования (возможность пересчёта бюллетеней);

– сокрытие результатов до окончания голосования (невозможность определения исхода до окончания голосования);

– решение голосующего не может быть тайно или явно кем-либо изменено (кроме, возможно, им самим). [2]

Также, как электронная система, она должна быть отказоустойчива в случае технических неисправностей (потеря электропитания), непреднамеренных (потеря избирателем ключа) и злоумышленных (намеренная выдача себя за другого избирателя, DoS/DDoS) атак.

## 1.3 Анализ существующих систем голосования

Системы голосования можно разделить на несколько типов:

- бумажную (традиционную);
- бумажно-электронную;
- электронную с прямой записью;
- электронную использующую публичные сети.

Обратим внимание на недостатки традиционной (бумажной) системы голосования. Главным недостатком является длительность подсчета голосов, заполнение соответствующих протоколов и т.д. При бумажном голосовании большое влияние на результат оказывает человеческий фактор – ошибки, возникающие как вследствие переутомления, недомогания и т.д., так и преднамеренные. К наиболее распространенным видам фальсификации можно отнести:

- подкуп избирателей;
- махинации со списком избирателей;
- вбрасывание в выносные урны фальшивых бюллетеней;

– подделка протокола;
– возможность использования «чистых» бюллетеней, не явившихся на избирательный участок граждан;
– порча бюллетеней.

Рассмотрим процесс голосования на традиционном участке, с урной и бумажными бюллетенями. В общем упрощенном виде он выглядит так: избиратель приходит на участок и предъявляет документ, удостоверяющий личность (паспорт). На участке работает участковая избирательная комиссия, член которой проверяет личность избирателя и наличие его в списке избирателей, который был составлен ранее. Если избиратель найден, член комиссии выдает избирателю бюллетень, а избиратель расписывается в получении бюллетеня. После этого избиратель отправляется в кабинку для голосования, заполняет бюллетень, и опускает его в урну. Чтобы все процедуры соблюдались строго по закону, за всем этим следят наблюдатели (представители кандидатов, общественных институтов наблюдения). После завершения голосования избирательная комиссия в присутствии наблюдателей производит подсчет голосов и устанавливает итоги голосования.

Бумажно-электронная система означает заполнение бумажных бюллетеней, а подсчет уже в электронном виде. Избиратель ставит отметку в бумажном бюллетене и вставляет его урну, в которой результат считывается с помощью сканера, а далее распознается. Обработка одного бюллетеня занимает несколько секунд. По окончанию времени голосования подсчитываются результаты по участку и протокол, который подписывается членами комиссии. Протоколы сохраняются на электронном носителе.

Система электронного голосования с прямой записью подразумевает использование избирателем электрооптических или механических компонентов для подачи своего голоса. Информация хранится на одном носителе и может передаваться на более высокие уровни избирательных комиссий. Такие системы применяются, в частности, в Нидерландах, США, Венесуэле и Бразилии. Отличие гибридной

системы голосования состоит в том, что информация хранится на отдельном устройстве.

В Финляндии электронное голосование проходит на избирательном участке, дистанционное голосование невозможно. Аутентификация и авторизация производится путем сканирования штрих-кода документа и сравнивания с электронным списком избирателей. При этом выдается информация о том, имеет ли право голоса данный субъект. Избирателю выдается карточка с электронным ключом, с помощью которой можно проголосовать. Для этого необходимо вставить ее в электронную урну и выбрать кандидата, данные которого отображены на экране и подтвердить выбор. После голосования карточка возвращается обратно организаторам голосования. Голос избирателя передается в избирательную комиссию. Проблема анонимности решается применением программы, отделяющей данные о пользователе от его голоса. Посмотреть результаты электронного голосования можно на официальном сайте сразу после окончания выборов.

США имеют наиболее длительный опыт использования электронных систем голосования, при этом на сегодняшний день правительство США продолжает усовершенствование аппаратного и программного обеспечения, из-за обширной критики, отвергающей подобные нововведения и реформирования избирательной системы. Вопрос демократии и прозрачности подсчета голосов занимает в политике США одно из важнейших мест. В большинстве штатов США впервые электронное голосование было применено на президентских выборах в 2000 году. В ходе этих выборов американскими специалистами был установлен высокий процент ошибок и сбоев у старых карточных автоматов. В 2002 году в США был принят закон «Акт содействия голосованию», установивший обязательное использование электронного голосования во всех штатах.

Системы электронного голосования, использующие публичные сети, применяют электронные бюллетени и не используют бумажные носители вовсе. Результаты голосования передаются по сетям. Примером таких систем является

голосование через Интернет и SMS. Информация может передаваться как по одному голосу, так и периодически набором голосов или по окончании времени голосования.

Для того, чтобы проголосовать, пользователю необходимо иметь «открытый» и «закрытый» ключи. «Открытый» используется для регистрации на сайте голосования, «закрытый» - как правило, для шифрования результата голосования.

В Эстонии Интернет-голосование было впервые применено на выборах в органы местного самоуправления в 2005 году. Голосование проходит через портал избирательной комиссии в Интернете с помощью карты, которая служит удостоверением личности как в банках, так и в государственных учреждениях: на ней хранится такая информация, как полное имя владельца, пол, национальный идентификационный номер, криптографические ключи и сертификаты. К карточкам предъявляются специальные требования, а также имеются особые требования к компьютерным операционным системам. [3]

В России был проведено несколько экспериментов по внедрению комплекса электронного голосования. Самой продвинутой системой является система «ГАС выборы», разработанная ЦИК России применяемая для выборов в государственную думу. Для включения в список участников необходимо подать заявление на портале Госуслуг. После получения заявления данные избирателя еще раз проходят проверку в ЦИК России и загружаются в компонент «Список избирателей» ПТК ДЭГ. При обращении на портал ДЭГ происходит его аутентификация и идентификация в списке избирателей, а также проверка того, что этот избиратель ранее еще не получал бюллетень. Для проведения идентификации и аутентификации используется ЕСИА Портала Госуслуг. Таким образом, сохраняется общая схема идентификации как при подаче заявления, так и при участии в голосовании. После этого начинается процедура анонимизации – избирателю выдается бюллетень, который не содержит никаких идентификационных отметок. В ПТК ДЭГ применяется криптографический алгоритм, известный в профессиональной среде как «слепая электронная подпись». Затем избиратель заполняет бюллетень, для этого пользователь

сначала переводится на другой домен – в анонимную зону. Перед переходом можно поднять VPN-соединение и сменить IP-адрес. На этом домене и происходит отображение бюллетеня и обработка выбора пользователя. Исходный код, который исполняется на устройстве пользователя, изначально открыт – его можно увидеть в браузере. После того как выбор сделан, бюллетень зашифровывается на устройстве пользователя с применением специальной схемы шифрования, отправляется и записывается в компонент «Распределенное хранение и подсчет голосов», построенный на базе блокчейн-платформы.

В таблице 1.1 сравним по параметрам существующие системы голосования. «+» – параметр реализован в системе, «-» – не реализован.

Таблица 1.1 – Сравнение существующих систем голосования по параметрам

| Параметр | Бумажное | Бумажно-электронное | Электронное с прямой записью | Электронное через публичные сети |
|---|---|---|---|---|
| Соответствует требованиям, предъявленным в разделе 1.2 | + | + | + | + |
| Автоматизированный подсчет голосов | - | + | + | + |
| Автоматизированный сбор голосов | - | - | + | + |
| Возможно проголосовать дистанционно | - | - | - | + |

Исходя из таблицы, так как система будет дистанционной и пользователь может быть где угодно, для передачи данных от пользователя к сервисам будем использовать публичные сети, это означает что все данные передаваемые должны подвергаться шифрованию.

## 1.4 Модель угроз и нарушителей безопасности информации

Основная особенность модели угроз и нарушителя безопасности информации в ПТК ДЭГ - учет не только угроз, характерных для информационных систем, но и специфических угроз, связанных с реализацией и использованием протоколов тайного дистанционного электронного голосования.

В соответствии с Приказом ФСТЭК России от 11.02.2013 № 17 «Об утверждении требований о защите информации, не составляющей государственную тайну, содержащейся в государственных информационных системах» для информации, обрабатываемой в ПТК ДЭГ, устанавливаются следующие классификационные признаки:

- высокий уровень значимости (УЗ-1);
- ПТК ДЭГ имеет федеральный масштаб так как функционирует на всей территории Российской Федерации.

В соответствии с постановлением Правительства Российской Федерации от 1 ноября 2012 г. № 1119 «Об утверждении требований к защите персональных данных при их обработке в информационных системах персональных данных» установлено, что ПТК ДЭГ актуален 3 тип угроз.

При разработке Модели угроз применялись методики, определённые в методическом документе ФСТЭК России «Методика определения угроз безопасности информации в информационных системах».

В таблице 1.2 определим возможные виды рисков и типовые негативные последствия от реализации угроз безопасности информации.

Таблица 1.2 – Виды рисков (ущерба) и типовые негативные последствия от реализации угроз безопасности информации

| № | Виды риска | Возможные последствия |
|---|---|---|
| У1 | Ущерб физическому лицу | Нарушение конфиденциальности (утечка) персональных данных.<br>«Травля» гражданина в сети «Интернет».<br>Разглашение персональных данных граждан |
| У2 | Риски юридическому лицу, индивидуальному предпринимателю | Нарушение законодательства Российской Федерации.<br>Нарушение штатного режима функционирования автоматизированноё системы управления и управляемого объекта и/или процесса<br>Потеря клиентов, поставщиков.<br>Потеря конкурентного преимущества. |
| У3 | Ущерб государству в области обеспечения обороны страны, безопасности правопорядка, социальной, политической, сферах деятельности | Нарушение выборного процесса.<br>Отсутствие доступа к государственной услуге.<br>Публикация недостоверной социально значимой информации на веб-ресурсах, которая может привести к социальной напряженности, панике среди населения и др.<br>Появление негативных публикаций в общедоступных источниках.<br>Доступ к системам и сетям с целью незаконного использования вычислительных мощностей.<br>Использование веб-ресурсов государственных органов для распространения и управления вредоносным программным обеспечением.<br>Утечка информации ограниченного доступа. Непредставление государственных услуг |

Определим объекты взаимодействия и виды воздействия на них таблице 1.3

Таблица 1.3 – Объекты воздействия и виды воздействия на них

| Негативные последствия | Объекты воздействия | Виды воздействия |
|---|---|---|
| Разглашение персональных данных граждан (У1) | База данных информационной системы, содержащая идентификационную информацию граждан | Утечка идентификационной информации граждан из базы данных |
| | Линия связи между сервером авторизации и обработки данных. | Перехват информации, содержащей идентификационную информацию и граждан, передаваемой по линиям связи. |
| | Приложение информационной системы, обрабатывающей идентификационную информацию граждан | Несанкционированный доступ к идентификационной информации граждан, содержащейся в приложении информационной системы |
| Непредставление государственных услуг (У3) | Приложение голосования | Отказ в обслуживании приложения |
| | Сервер баз данных портала государственных услуг | Отказ в обслуживании сервера управления базами данных |
| | | Подмена информации в базах данных на недостоверную |
| | | Утечка персональных данных граждан |

С учетом наличия прав доступа и возможностей по доступу к информации и/или к компонентам информационной системы нарушители подразделяются на два типа:

1) внешние нарушители (тип I) – лица, не имеющие права доступа к информационной системе, ее отдельным компонентам и реализующие угрозы безопасности информации из-за границ информационной системы;

2) внутренние нарушители (тип II) – лица, имеющие право постоянного или разового доступа к информационной системе, ее отдельным компонентам.

В зависимости от потенциала, требуемого для реализации угроз безопасности информации, нарушители подразделяются на:

– нарушителей, обладающих базовым (низким) потенциалом нападения при реализации угроз безопасности информации в информационной системе;

– нарушителей, обладающих базовым повышенным (средним) потенциалом нападения при реализации угроз безопасности информации в информационной системе;

– нарушителей, обладающих высоким потенциалом нападения при реализации угроз безопасности информации в информационной системе.

В таблице 1.4 определим виды нарушителей безопасности информации

Таблица 1.4 – Типы и виды нарушителей безопасности информации

| Тип нарушителя | Вид нарушителя |
|---|---|
| Внешний | Специальные службы иностранных государств (блоков государств |
| | Террористические, экстремистские группировки. |
| | Преступные группы (криминальные структуры); Внешние субъекты (физические лица); |
| | Разработчики, производители, поставщики программных, технических и программно-технических средств |

Продолжение таблицы 1.4

| Тип нарушителя | Вид нарушителя |
|---|---|
| Внешний | Лица, привлекаемые для установки, наладки, монтажа, пусконаладочных и иных видов работ |
| | Конкурирующие организации |
| | Авторизованные пользователи систем и сетей |
| | Лица, обеспечивающие поставку, сопровождение и ремонт технических средств ПТК ДЭГ |
| Внутренний | Пользователи ПТК ДЭГ |
| | Бывшие работники |
| | Администраторы ПТК ДЭГ |
| | Лица, привлекаемые для установки, наладки, монтажа, пусконаладочных и иных видов работ |
| | Обслуживающий персонал |

При определении источников угроз безопасности информации необходимо исходить из предположения о наличии повышенной мотивации внешних и внутренних нарушителей, преднамеренно реализующих угрозы безопасности информации.

Кроме того, необходимо учитывать, что такие виды нарушителей как специальные службы иностранных государств и террористические, экстремистские группировки могут привлекать (входить в сговор) внутренних нарушителей, в том числе обладающих привилегированными правами доступа. В этом случае уровень возможностей актуальных нарушителей будет определяться совокупностью возможностей нарушителей, входящих в сговор.

В таблице 1.5 рассмотрим возможную мотивация рассмотренных выше нарушителей.

Таблица 1.5 – Возможные цели реализации угроз безопасности информации нарушителями

| Виды нарушителя | Возможные цели реализации угроз безопасности информации |
|---|---|
| Специальные службы иностранных государств | Нанесение ущерба государству в области обеспечения обороны, безопасности и правопорядка, а также в иных отдельных областях его деятельности или секторах экономики, в том числе дискредитация или дестабилизация деятельности отдельных органов государственной власти, организаций, получение конкурентных преимуществ на уровне государства, срыв заключения международных договоров, создание внутриполитического кризиса |
| Террористические, экстремистские группировки | Нанесение ущерба отдельным сферам деятельности или секторам экономики государства.<br>Дестабилизация общества. Дестабилизация деятельности органов государственной власти, организаций |
| Преступные группы (криминальные структуры)<br>Отдельные физические лица | Получение финансовой или иной материальной выгоды.<br>Желание самореализации |

Продолжение таблицы 1.5

| Виды нарушителя | Возможные цели реализации угроз безопасности информации |
|---|---|
| Разработчики программных, программно-аппаратных средств | Внедрение функциональных программные аппаратные средства на этапе разработки.<br>Получение конкурентных преимуществ.<br>Получение финансовой или иной материальной выгоды.<br>Непреднамеренные, неосторожные или неквалифицированные действия |
| Лица, обеспечивающие поставку программных, программно- аппаратных средств, обеспечивающих систем<br>Лица, привлекаемые для установки, настройки, испытаний, пусконаладочных и иных видов работ | Получение финансовой или иной материальной выгоды.<br>Непреднамеренные, неосторожные или неквалифицированные действия.<br>Получение конкурентных преимуществ |
| Авторизованные пользователи систем и сетей<br>Системные администраторы и администраторы безопасности | Получение финансовой или иной материальной выгоды.<br>Любопытство или самореализации.<br>Месть за ранее совершенные действия.<br>Непреднамеренные, неосторожные или неквалифицированные действия. |

Организационные меры и средства защиты информации, применяемые в ПТК, должны обеспечивать защиту от угроз безопасности информации, связанных с действиями нарушителей с высоким потенциалом.

В качестве исходных данных для определения угроз безопасности информации использовался банк данных угроз безопасности информации (bdu.fstec.ru)

Рассматриваются угрозы:

– угроза внедрения кода или данных (УБИ. 006);

– угроза восстановления и/или повторного использования аутентификационной информации (УБИ. 008);

– угроза использования информации идентификации/аутентификации, заданной по умолчанию (УБИ. 030);

– угроза несанкционированного доступа к аутентификационной информации (УБИ. 074);

– угроза несанкционированного изменения аутентификационной информации (УБИ. 086);

– угроза обхода некорректно настроенных механизмов аутентификации (УБИ. 100);

– угроза перехвата данных, передаваемых по вычислительной сети (УБИ. 116);

– угроза удаления аутентификационной информации (УБИ. 152).

Также в ПТК ДЭГ рассматриваются угрозы, связанные с использованием протоколов голосования. К данным угрозам относятся:

- возможность со стороны нарушителя, используя ПО и технологические решения ПТК ДЭГ извлечь сведения о выборе избирателя, группы избирателей, всех избирателей, а также идентифицировать избирателя по выбору;
- возможность реализации голосования более одного раза;
- подмена голосов избирателей;
- некорректная запись голоса избирателя;

– досрочное прекращение голосования;
– деанонимизация избирателя;
– установление промежуточных итогов голосования до его завершения.

В составе ПТК ДЭГ необходимо использовать сертифицированные по требованиям безопасности информации средства защиты информации:

– средства защиты информации не ниже 4 класса и соответствующие 4 уровню доверия;
– средства контроля съемных машинных носителей информации не ниже 4 класса;
– средства вычислительной техники не ниже 5 класса;
– системы обнаружения вторжений не ниже 4 класса;
– средства антивирусной защиты не ниже 4 класса;
– средства межсетевого экранирования не ниже 4 класса;
– средства доверенной загрузки не ниже 4 класса.

В ПТК ДЭГ предполагаемый к использованию класс криптографической защиты для нейтрализации угроз безопасности информации при передаче персональных и иных данных по каналам связи между ЦОД ПТК ДЭГ определен как КА.

Для реализации подсистемы подключения пользователей к порталам ЕПГУ и ПТК ДЭГ для авторизации пользователей и получения бюллетеня голосования предполагаемый к использованию класс криптографической защиты для серверной компоненты класс СКЗИ определен как КС3.

Предполагаемый к использованию класс криптографической защиты в сегменте пользователей ПТК ДЭГ (избиратель) для подключения пользователей к порталам ЕПГУ и ПТК ДЭГ, авторизации пользователей и получения бюллетеня голования, для нейтрализации угроз безопасности информации при передаче персональных данных по каналам связи, а также наложения и проверки ЭП определен как КС1.

Предполагаемый к использованию класс криптографической защиты на стороне администраторов управления, председателей и членов ИК ДЭГ (председатель

ИК ДЭГ, оператор ИК ДЭГ, администраторы ИТ, администраторы ИБ), при взаимодействии с ПТК ДЭГ по каналам связи выходящими за пределы ЦОД, ввиду регулярного характера взаимодействия с системой и категории обрабатываемых данных (управляющая информация) определен как КА.

Предполагаемый к использованию класс криптографической защиты на стороне администраторов управления, председателей и членов ИК ДЭГ (председатель ИК ДЭГ, оператор ИК ДЭГ, администраторы ИТ, администраторы ИБ), при взаимодействии с ПТК ДЭГ по каналам связи не выходящими за пределы контролируемой зоны ЦОД, ввиду регулярного характера взаимодействия с системой и категории обрабатываемых данных (управляющая информация) определен как КС3.

Предполагаемый к использованию класс криптографической защиты для ключевого центра определен как класс СКЗИ КА.

При разработке защищенного веб-приложения для электронного голосования необходимо руководствоваться моделями угроз и нарушителя, так как с их помощью удастся построить качественную систему защиты.

## 1.5 Выводы по разделу

В первом разделе был определен объект разработки, определены требования к ДЭГ. Произведен сравнительный анализ существующих систем голосования, в результате анализа делаем вывод, что из существующих систем голосования, можем использовать технологию голосования через публичные сети, так как только она обеспечивает возможность проголосовать дистанционно.

Спрогнозированы угрозы и уязвимости разрабатываемой системы и рассмотрены способы их предотвращения. Также была разработана модель потенциального нарушителя информационной безопасности веб-приложения для электронного голосования. В составе ПТК ДЭГ необходимо использовать сертифицированные по требованиям безопасности информации средства защиты информации средства защиты информации не ниже 4 класса и соответствующие 4 уровню доверия.

Для ПТК ДЭГ необходимо обеспечить выполнения требований, предъявляемых к 1 (первому) классу защищенности информационных систем.

В ПТК ДЭГ необходимо обеспечить третий уровень защищенности персональных данных при их обработке в ПТК ДЭГ (УЗ-3).

# 2 Разработка технического задания

## 2.1 Постановка задачи

В данной главе необходимо проработать технические решения для разработки системы голосования, выбрать из существующих протоколов тайного голосования или разработать собственный в соответствии с требованиями, поставленными в главе 1

Необходимо разработать концепцию модулей системы в соответствии с протоколом и требованиями к системе. Спланировать архитектуру разрабатываемого веб-приложения и отобразить принцип взаимодействия пользователя с системой.

Учитывая эти сведения, нужно определить какая техническая база будет использоваться при разработке приложения.

## 2.2 Сравнительный анализ протоколов электронного голосования

Целью данной главы является выбор протокола голосования, который отвечает требованиям выставленный нами в разделе 1.3

Рассмотрим алгоритм простого протокола электронного голосования по:

Шаг 1. Агентство, проводящее электронное голосование (далее А) выкладывает списки возможных участников выборов.

Шаг 2. Участник, допущенный к выборам (далее В) сообщает о своем намеренье участвовать в голосовании.

Шаг 3. А выкладывает списки зарегистрированных В.

Шаг 4. А создает закрытый ($К_{А_{зак}}$) и открытый ($К_{А_{отк}}$) ключ и выкладывает в общий доступ $К_{А_{отк}}$, чтобы любой мог зашифровать сообщение, но расшифровать мог только А.

Шаг 5. В создает свои ключи $К_{В_{зак}}$ и $К_{В_{отк}}$ и выкладывает в общий доступ $К_{В_{отк}}$, чтобы любой мог проверить его электронный избирательный бюллетень (далее С), но подписать мог только он сам.

Шаг 6. В формирует сообщение С, где выражает свой выбор, подписывает $К_{В_{зак}}$, шифрует $К_{А_{отк}}$ и отправляет А.

Шаг 7. А собирает С, расшифровывает с помощью $К_{В_{отк}}$ и публикует подсчитанные результаты.

Довольно простой протокол, помогает защититься от подделки голосов и внешнего вмешательства, но В должен доверять А, чья работа никем не контролируется. [4]

Далее рассмотрим алгоритм протокола Нурми-Салома-Сантина или, другими словами, протокола двух агентств:

Шаг 1. Валидатор (далее V) отправляет секретные опознавательные метки (далее М) всем В до голосования.

Шаг 2. V отправляет А весь набор М, но без информации о том, кому они принадлежат.

Шаг 3. В создает свои ключи $К_{В_{зак}}$, $К_{В_{отк}}$ и выкладывает в общий доступ $К_{В_{отк}}$, а также создает секретный ключ ($К_{В_{сек}}$), который нужен, чтобы никто не узнал содержимое бюллетеня до нужного момента.

Шаг 4. В формирует сообщение С, где выражает свой выбор, подписывает $К_{В_{зак}}$, прикладывает к нему полученную М и шифрует $К_{В_{сек}}$.

Шаг 5. К зашифрованному тексту В прикладывает М и отправляет А.

Шаг 6. А получает зашифрованный текст, по М определяет, что он пришел от В, но не знает от кого именно и как В проголосовал, после публикует его.

Шаг 7. Опубликованный зашифрованный текст служит информацией, чтобы В отправил $К_{В_{сек}}$.

Шаг 8. А собирает ключи, расшифровывает текст, подсчитывает голоса и присоединяет к опубликованному зашифрованному тексту С без М.

На 6 шаге А не сможет отрицать, что не получал сообщения от В. Благодаря публикации зашифрованного текста и бюллетеня, каждый В может проверить, что его голос был учтен должным образом.

Данный протокол имеет минусы, если А вступит в тайный сговор с V, то он сможет манипулировать голосованием, специально не принимая сообщения от некоторых В. Так же присутствует проблема «мертвых душ», если V специально внесет несуществующих В, то А сможет фальсифицировать бюллетени от них. [5]

Следующим рассмотрим алгоритм протокола Фудзиока-Окамото-Охта. Частично решает проблему сговора двух агентств. Работа протокола заключается в заранее выбранном способе маскирующего шифрования – это особый вид шифрования, который позволяет убедиться, что документ подлинный и был подписан авторизированным пользователем, но не дает информации о содержащихся данных. Алгоритм выглядит следующим образом:

Шаг 1. V утверждает список В.

Шаг 2. В создает свои ключи $К_{В_{зак}}$, $К_{В_{отк}}$, $К_{В_{сек}}$ и выкладывает в общий доступ $К_{В_{отк}}$.

Шаг 3. В формирует сообщение С, где выражает свой выбор, шифрует его $К_{В_{сек}}$, маскирует, подписывает $К_{В_{зак}}$ и отправляет V.

Шаг 4. V создает свои ключи $К_{V_{зак}}$, $К_{V_{отк}}$ и выкладывает в общий доступ $К_{V_{отк}}$.

Шаг 5. V удостоверяется, что С принадлежит В, который еще не голосовал, подписывает его $К_{V_{зак}}$ и отправляет В.

Шаг 6. В удаляет слой маскирующего шифрования и отправляет сообщение С к А.

Шаг 7. А проверяет подписи В и V и помещает зашифрованный С в специальный список, который будет опубликован после голосования.

Шаг 8. После публикации списка, В отправляет А свой $К_{В_{сек}}$.

Шаг 9. А собирает ключи, расшифровывает С и подсчитывает голоса, при этом публикует декодирующие ключи вместе с зашифрованными С, чтобы В смогли самостоятельно проверить результаты голосования.[6]

В 1996 году было предложено модифицировать протокол, который получил название Sensus. Дополнением является то, что после помещения зашифрованного С в специальный список А отправляет подписанный С обратно В в качестве квитанции, таким образом можно закончить голосование в течении одного сеанса, что необходимо при дистанционном голосовании, ведь по окончании голосования не всегда возможно будет связаться с устройством пользователя. Это удобно для конченого пользователя и дает дополнительные гарантии участия в выборах. Если А вступит в тайный сговор с V, то уже не сможет опознать В до получения ключа. Остается лишь проблема подачи голосов за В, не пришедших на выборы. [7]

Рассмотри алгоритм протокола He-Su. Данная схема решает проблему тайного сговора А и V. Как в предыдущих протоколах используется идея слепой подписи, но подписывается открытый ключ В, а не его бюллетень. Это позволяет скорректировать свой голос до окончания голосования. Алгоритм:

Шаг 1. V утверждает список В, создает свои ключи $К_{V_{зак}}$, $К_{V_{отк}}$ и выкладывает в общий доступ $К_{V_{отк}}$.

Шаг 2. В создает свои ключи $К_{В_{зак}}$, $К_{В_{отк}}$, генерирует случайное число R вычисляет хэш-функцию h от $К_{В_{отк}}$, маскирует ее R и отправляет V полученную функцию, которая выглядит следующим образом: $f = К_{V_{отк}}(R) \cdot h(К_{В_{отк}})$.

Шаг 3. V удостоверяется в праве В голосовать, подписывает полученное сообщение, т.е. $К_{V_{зак}}(К_{V_{отк}}(R) \cdot h(К_{В_{отк}})) = R \cdot К_{V_{зак}}(h(К_{В_{отк}}))$ и отправляет В.

Шаг 4. В удаляет слой маскирующего шифрования, проверяет подлинность подписи V, т. е. $К_{V_{отк}}(К_{V_{зак}}(h(К_{В_{отк}}))) = h(К_{В_{отк}})$ и отправляет А $К_{В_{отк}}$ и подпись V, т.е. $К_{V_{зак}}(h(К_{В_{отк}}))$.

Шаг 5. А проверяет подлинность подписи V, проверяет совпадение хэш-функции от $К_{В_{отк}}$ с той, что хранится в подписи V, добавляет $К_{В_{отк}}$ в список авторизованных ключей и сообщает об этом В.

Шаг 6. В формирует сообщение С, где выражает свой выбор, шифрует его созданным $К_{В_{сек}}$ и отправляет А набор состоящий из $К_{В_{отк}}$, зашифрованного $К_{В_{сек}}$ сообщение С и зашифрованную $К_{В_{зак}}$ хэш-функцию от зашифрованного$К_{В_{сек}}$ со- общения С.

Шаг 7. А проверяет $К_{В_{отк}}$ со списком, созданным ранее, сравнивает хэш-функцию сообщения С зашифрованного $К_{В_{сек}}$ и хэш-функцию, полученную при помощи $К_{В_{зак}}$ и публикует весь набор в открытом списке.

Шаг 8. После публикации списка В отправляет А новый набор состоящий из $К_{В_{отк}}$, $К_{В_{сек}}$ и зашифрованную $К_{В_{зак}}$ хэш-функцию от $К_{В_{сек}}$.

Шаг 9. А проверяет подлинность $К_{В_{сек}}$, сравнивая хэш-функцию от $К_{В_{сек}}$ и хэш-функцию полученную при помощи $К_{В_{зак}}$, если все верно, то расшифровывает полученную ранее С, публикует все данные и подсчитывает голоса.

Шаг 10. После голосования V публикует утвержденный список В, а А – список авторизованных ключей. [8]

А и V не могут тайно сговориться, потому что публикуют списки, поэтому нельзя внести несуществующих избирателей и проголосовать за не пришедших. Минусами является уязвимость перед DoS-атаками, так как требуется большое количество ресурсов для поддержания работоспособности протокола из-за его сложности.

В соответствии с указанными преимуществами и недостатками, наиболее подходящим для дистанционного голосования является протокол Фудзиока-Окамото-Охта.

## 2.3 Разработка концепции модулей системы голосования

При бумажном голосовании тайна голосования обеспечивается физическим разрывом между двумя местами — местом, где избиратель удостоверяет своё право голосовать, и местом, где он отдаёт голос. В первом месте — это столик

избирательной комиссии участка — избиратель идентифицируется по паспорту и ему выдаётся анонимизированный бюллетень. Во втором месте — урне для голосования — сам факт наличия бюллетеня является подтверждением права на голосование, личность избирателя уже неважна и, собственно, неизвестна.

В большинстве систем электронного голосования, этого разрыва нет: аутентификация и голосование проходят на одном и том же сервере, находящемся под контролем одних и тех же людей. Каковые, разумеется, могут иметь собственные политические интересы и, соответственно, быть потенциально нечистоплотными на руку.

В ДЭГ можно реализовать такой физический разрыв с помощью разделения системы на два разных сервера.

Сервис регистратор пользователей проверяет, может ли данный пользователь голосовать, а сервис учета голосов – производит учет и подсчет голосов

Сервис регистратор хранит в себе списки с идентификаторами пользователей, а также их публичные ключи.

Аутентификацией и авторизацией пользователей занимается сторонняя система, которой доверяют проводящие голосование (например ЕСИА), чтобы сама система электронного голосования могла быть использована в любых видах голосования с подключением к существующим системам. В случае использования системы голосования в отрыве от других систем, сервис регистратор будет иметь в себе модуль регистрации, но хранить в себе будет только логин и хеш от пароля пользователя, то есть не хранить персональные данные пользователя, в рамках проведения голосования – это не нужно.

В зависимости от целей и важности голосования, аутентификация может проводиться:

– Парой логин-пароль или PIN-кодом по SMS (например, соцопросы или решение локальных вопросов городского хозяйства)

– По номеру партбилета пользователя, включая электронный партбилет на базе NFC/RFID (например, текущие внутрипартийные голосования)

– По аутентификации в ЕСИА (внутрипартийные праймериз, внепартийные голосования, включая общегосударственные выборы и референдумы)

ЕСИА — Единая система идентификации и аутентификации — это система авторизации в «Госуслугах»

Отметим, что биометрические датчики (датчик отпечатка глаза, радужки глаза и т.п.) использоваться для аутентификации в электоральных системах не могут, т.к. не отдают наружу собственно биометрические данные, а лишь подтверждают, что данное лицо является владельцем данного смартфона. Владелец пяти смартфонов, соответственно, сможет аутентифицироваться пять раз. Эти датчики могут использоваться лишь для подтверждения доступа к приложению, используемому для голосования, чтобы посторонний человек, получивший доступ к смартфону, не отдал голос за его владельца.

Использование биометрических данных для аутентификации в системе голосования потенциально возможно, но лишь в случае добровольного предоставления их пользователями и обработки со стороны сервера аутентификации пользователей — например, по фотографии лица.

При успешном прохождении аутентификации и авторизации, сервис аутентификации выдает регистратору только уникальный идентификатор пользователя в любом формате. Хеш от этого идентификатора попадает в список голосующих. Таким образом сервер регистратор не хранит в себе конфиденциальных данных голосующих. Даже в случае раскрытия списка голосующих, получить по хешу идентификатор пользователя в системе аутентификации, а по нему получить персональные данные человека очень трудно.

Сервис учета голосов хранит в себе список с зашифрованными бюллетенями во время голосования, а также на фазе отправке приватных ключей и список с приватными ключами пользователей. База данных с зашифрованными бюллетенями периодически реплицируется и отправляется на устройства наблюдателей. В конце голосования, сервис получает приватные ключи от пользователей, когда уже нельзя повлиять на результат голосования, отправляет их наблюдателям вместе с

итоговым списком бюллетеней. Расшифровывает бюллетени и производит подсчет голосов.

Оба сервиса разворачивается с помощью платформы HEROKU, которая обеспечивает шифрование HTTP трафика с помощью SSL, чтобы защититься от MITM-атак. Хоть и ход голосования таким образом не узнать, ведь бюллетени передаются зашифрованными, но, если перехватить передачу бюллетеня, а потом передачу ключа дешифрования, можно выяснить за какого кандидата проголосовал конкретный человек, а это уже нарушение тайны голосования. Так же это защитит от перехвата данных для авторизации.

Приложение – клиент голосующего, генерирует хранит в себе публичный и приватный ключ голосующего.

В ходе голосования наблюдатели получают копии списка бюллетеней через равные промежутки времени, и могут сравнивать реплики между собой, например, что бюллетени в прошлой реплике остались прежними в текущей. Так же по списку бюллетеней можно вести подсчет сколько участников уже проголосовало и исследовать количество голосов по времени. Но наблюдатели до окончания голосования не могут увидеть, что находятся в этих бюллетенях, они только видят их количество, так как бюллетени зашифрованы.

Технические специалисты могут следить за работой сервисов: смотреть сколько и какие бюллетени поступили на вход сервису учета голосов, сколько и какие были приняты, если есть непринятые, то по какой причине, есть ли разница в количестве прошедших авторизацию и в количестве бюллетеней, отправились ли голосующему подписанные бюллетени.

Наблюдатели следят за ходом голосования как вручную, проверяя копии базы данных пришедшие к ним, но и автоматически с помощью модуля аудита, который при получении новой копии, будет проверять на целостность и верность удостоверяющих подписей. Ручная проверка наблюдателей необходима, так как модуль аудита тоже необходимо контролировать на предмет ошибок и компрометации злоумышленниками.

В момент окончания голосования и публикации его результатов приватные ключи для расшифровки списка бюллетеней, которые реплицировались в ходе голосования рассылается наблюдателям, так что они могут самостоятельно подсчитать результат голосования и сравнить его с опубликованным — это сделает невозможной подмену результата.

Кроме того, наблюдатели могут сверить число голосов, зарегистрированных сервисом учёта голосов, с числом избирателей, зарегистрированных сервисом регистрации, чтобы исключить вариант вброса анонимных голосов владельцами сервера учёта голосов.

При использовании протокола Фудзиока-Окамото-Охта сервис регистратор может произвести голосование за участников, которые не пришли на выборы, но из-за того, что приватные ключи остаются у голосующих до окончания голосования, увидеть результаты голосования в его ходе невозможно. Так что голосование за кандидата будет заметно на статистике. Так же сервис не знает заранее, кто из голосующих не придет на выборы, а всплеск голосований под конец выборов будет заметен.

## 2.4 Выводы по разделу

В данном разделе были проработаны технические решения для разработки системы дистанционного электронного голосования.

Для реализации системы дистанционного электронного голосования выберем протокол Фудзиока-Окамото-Охта. Так как он отвечает требованиям, предъявленным в главе 1.3. А также является самым подходящим протоколом с учетом большого количества устройств с различными вычислительными способностями и качеством соединения. В соответствии с этим протоколом голосование состоит из нескольких этапов:

– утверждение списков избирателей;

– голосование (подпись бюллетеней пользователем, регистратором, прием сервисом учета голосов);
– подтверждение голосов (передача пользователями приватных ключей сервису учета голосов);
– подсчет голосов.

Система голосования представляет собой сервис регистратор, сервис учета голосов, систему аудита и клиентское приложение.

# 3 Разработка системы дистанционного электронного голосования

## 3.1 Постановка задачи

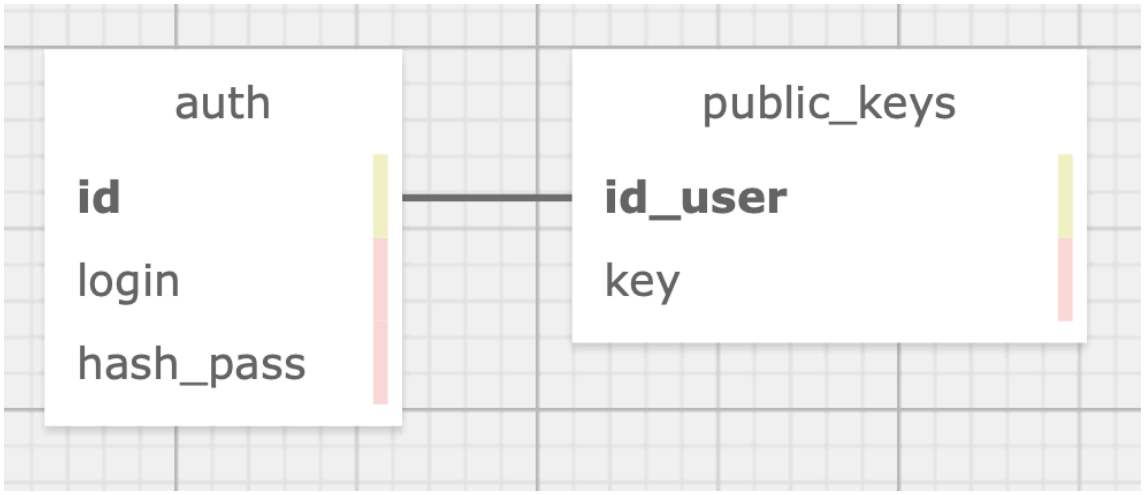
В данной главе необходимо разработать систему дистанционного электронного голосования (написать исходный код, спроектировать базу данных) в соответствии с техническими решениями, представленными в главе 2.

## 3.2 Разработка сервиса регистратора

Начнем разработку с сервиса регистратора. Для начала необходимо спроектировать базу данных сервиса. По техническому заданию, в ней будут храниться идентификаторы голосующих и их публичные ключи. Так же заложим сюда модуль авторизации и регистрации, для случая, если систему голосования не будут использовать уже с существующей системой авторизации, например ЕСИА.

В соответствии с входными данными в качестве СУБД используется PostgreSQL. На рисунке 3.1 изображена схема базы данных.

Рисунок 3.1 – Схема базы данных сервиса авторизации

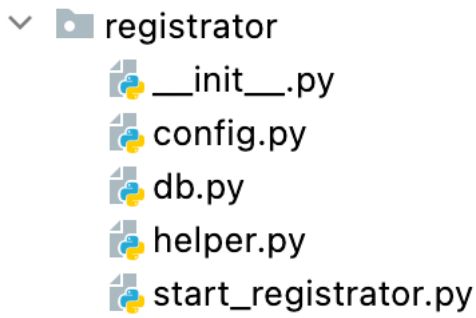



Созданная база данных registrator имеет две таблицы:

1) auth – таблица для аутентификации пользователей, которая используется, если нет стороннего сервиса решающего эту задачу. Содержит в себе следующие данные:
   - id – идентификатор пользователя;
   - login – логин пользователя;
   - hash_pass – хеш от пароля.
2) public_keys – таблица для хранения публичных ключей голосующих. Содержит в себе следующие данные:
   - id_user – хеш от идентификатора пользователя в таблице auth или хеш от идентификатора пользователя в сторонней системе аутентификации;
   - key – маскированный публичный ключ пользователя.

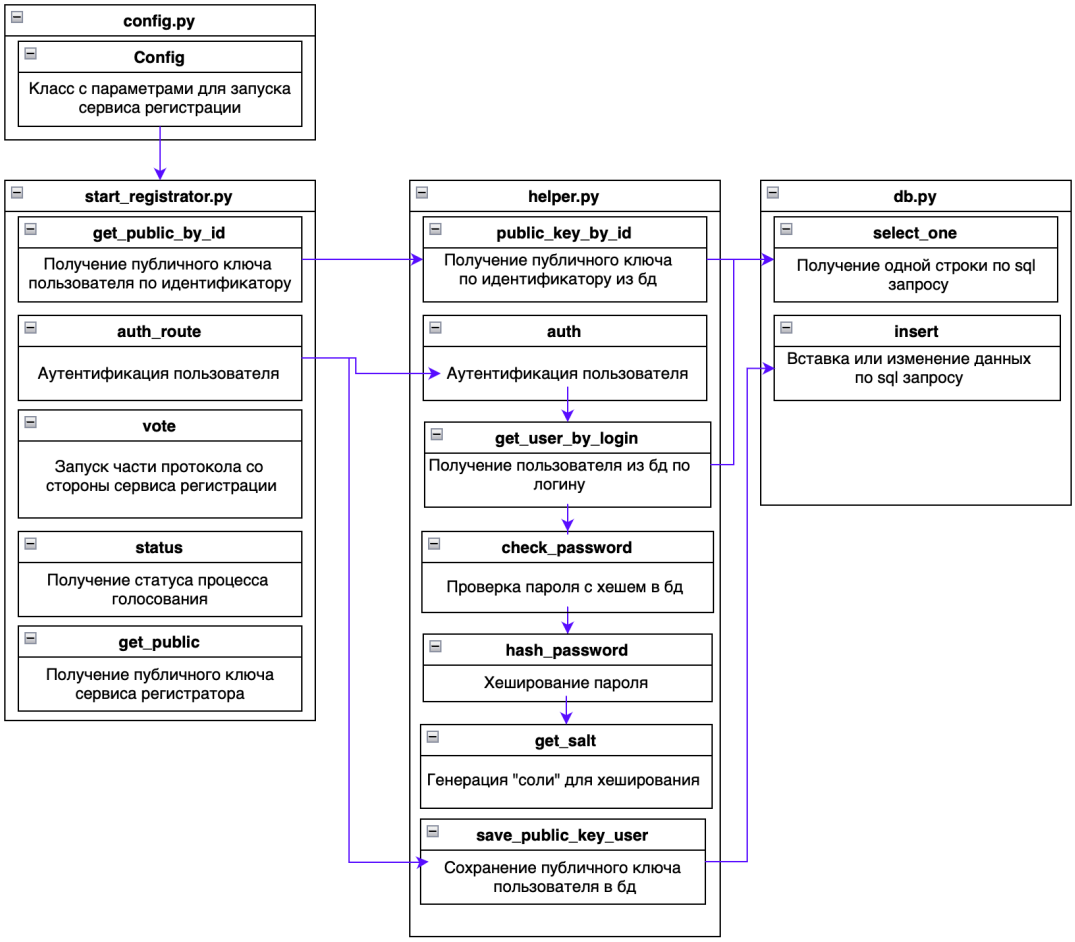
Далее перейдем к коду реализации сервиса. Общая структура представлена на рисунке 3.2

Рисунок 3.2 – Общая структура файлов сервиса регистрации




Входной точкой и скриптом запуска является файл start_registrator.py. Он отвечает за старт сервиса и ожидает подключения по https. Параметры запуска сервиса хранятся в файле config.py. В процессе обработки запросов на сервисе выполняется бизнес-логика, хранящаяся в файле helper.py, который в свою очередь подключается к базе данных, за взаимодействие с которой отвечает файл db.py. Логическая связь скриптов и функций в них отображено на рисунке 3.3

Рисунок 3.3 – Логическая связь методов сервиса регистратора

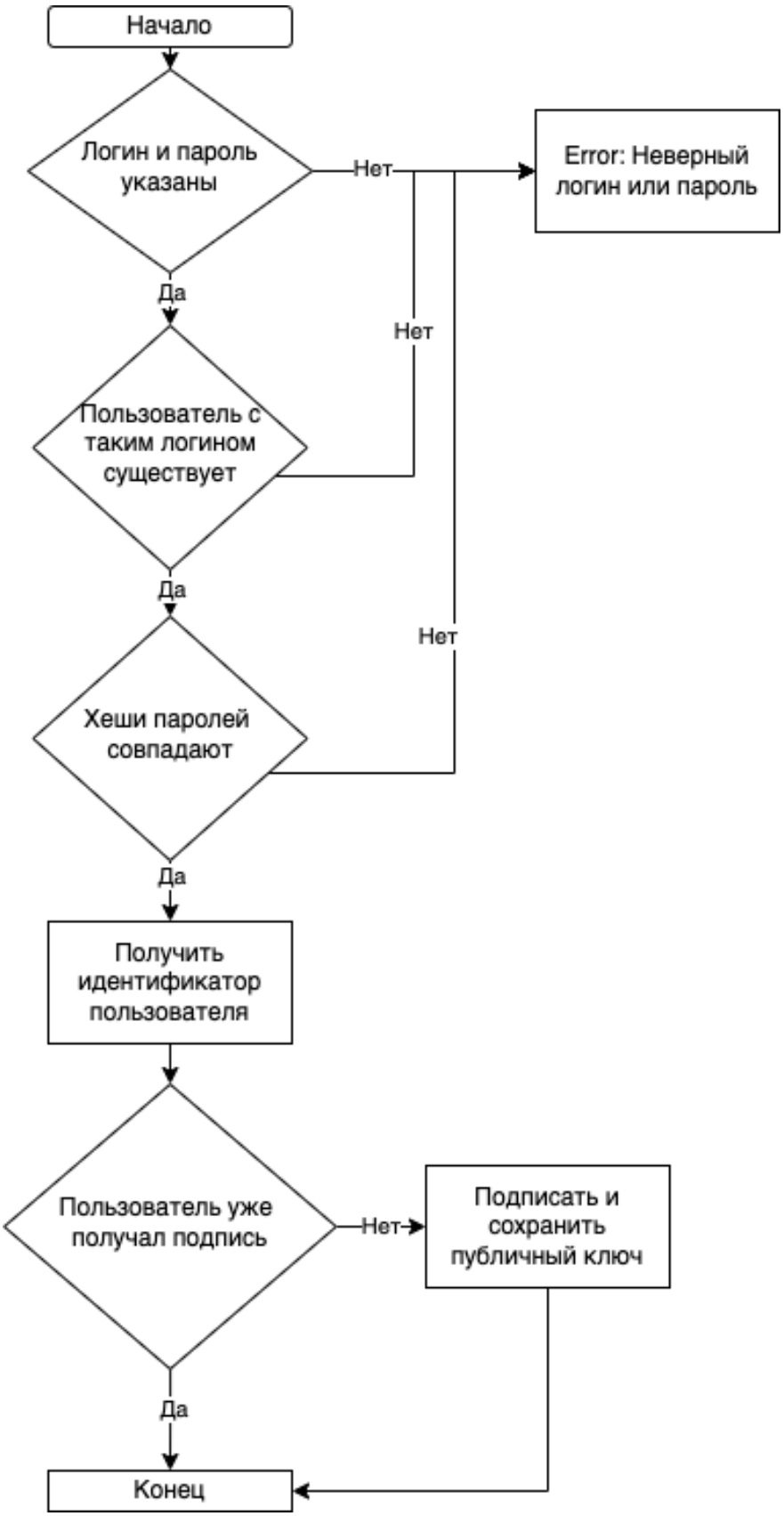



Сервис регистрации может сделать 5 действий:

- авторизовать пользователя и сохранить его публичный ключ;
- подписать ключ;
- предоставить статус голосования;
- предоставить публичный ключ сервиса
- предоставить маскированный публичный ключ пользователя.

Рассмотрим алгоритмы каждого действия в блок-схемах. Алгоритм работы пункта 1 и 2 отображен на рисунках 3.4 – 3.5.

Рисунок 3.4 – Блок схема авторизации в сервисе регистрации



При получении статуса голосования сервис регистрации отправляет данные о дате и времени начала и конца фаз голосования, а именно:

- дата и время начала голосования;
- дата и время начала этапа подтверждения голосов;
- дата и время завершения процесса голосования.

До тех по пока не настало время этапа подтверждения голосов сервис регистрации подписывает бюллетени. В остальное же время это не происходит.

Публичный ключ сервиса генерируется каждый раз новый при перезапуске и хранится в оперативной памяти компьютера.

Публичный ключ пользователя достается из бд по идентификатору с помощью sql запроса.

Полный листинг исходного кода находится в приложении А.

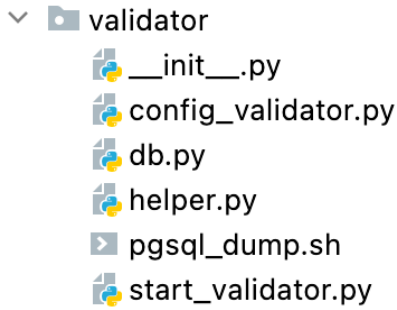
### 3.3 Разработка сервиса учета голосов

Так же начнем разработку со схемы базы данных. Созданная база данных validator имеет одну таблицу bulletins. Содержит в себе следующие данные:

- id – идентификатор бюллетеня;
- message – зашифрованный бюллетень;
- private_key – ключ для дешифрования бюллетеня.
- sign_user – подпись пользователя
- sign_registrator – подпись регистратора
- date_time – дата время принятия бюллетеня

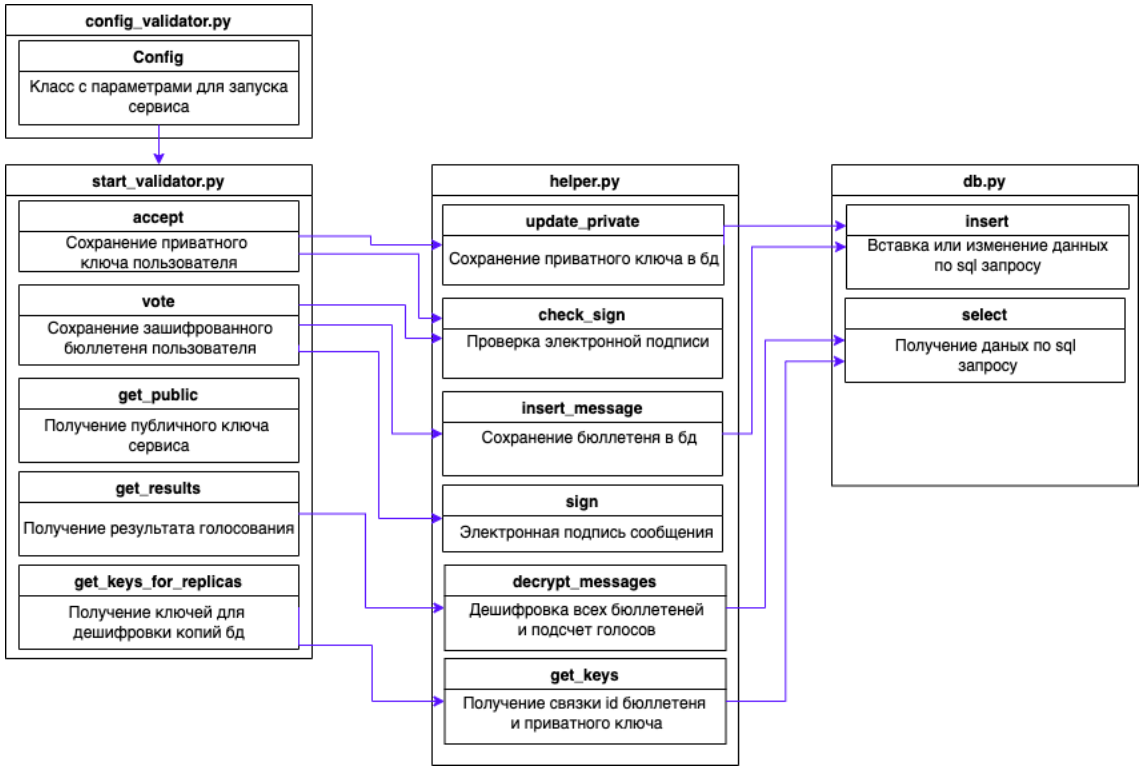
Далее перейдем к коду реализации сервиса. Общая структура представлена на рисунке 3.5

Рисунок 3.5 – Структура сервиса учета голосов




Входной точкой и скриптом запуска является файл start_validator.py. Он отвечает за старт сервиса и ожидает подключения по https. Параметры запуска сервиса хранятся в файле config_validator.py. В процессе обработки запросов на сервисе выполняется бизнес-логика, хранящаяся в файле helper.py, который в свою очередь подключается к базе данных, за взаимодействие с которой отвечает файл db.py. Логическая связь скриптов и функций в них отображено на рисунке 3.6

Рисунок 3.6 – Логическая связь методов сервиса учета голосов



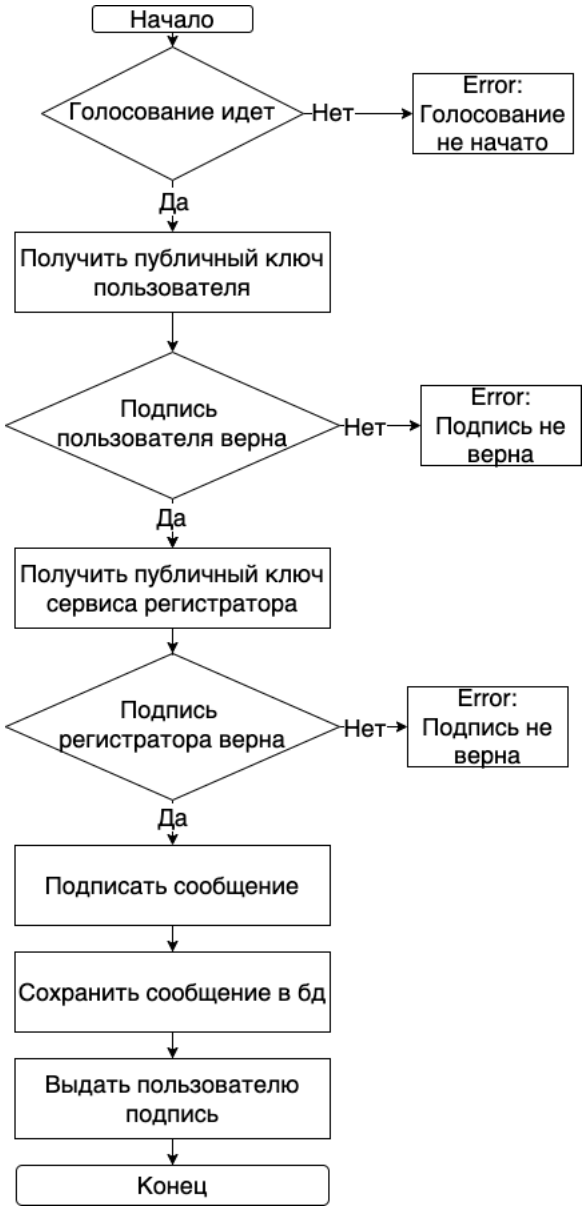

Сервис учета голосов может сделать следующие действия:

- сохранить и подписать зашифрованный бюллетень;
- сохранить приватный ключ для дальнейшей дешифровки бюллетеней;

– дешифровать все бюллетени и подсчитать итого;
– создать и выдать зашифрованную копию базы данных;
– выдать ключи для дешифровки копий базы данных.

Рассмотрим алгоритмы принятия бюллетеня в виде блок схемы, рисунок 3.7

Рисунок 3.7 – Блок схема принятия бюллетеня сервисом учета голосов

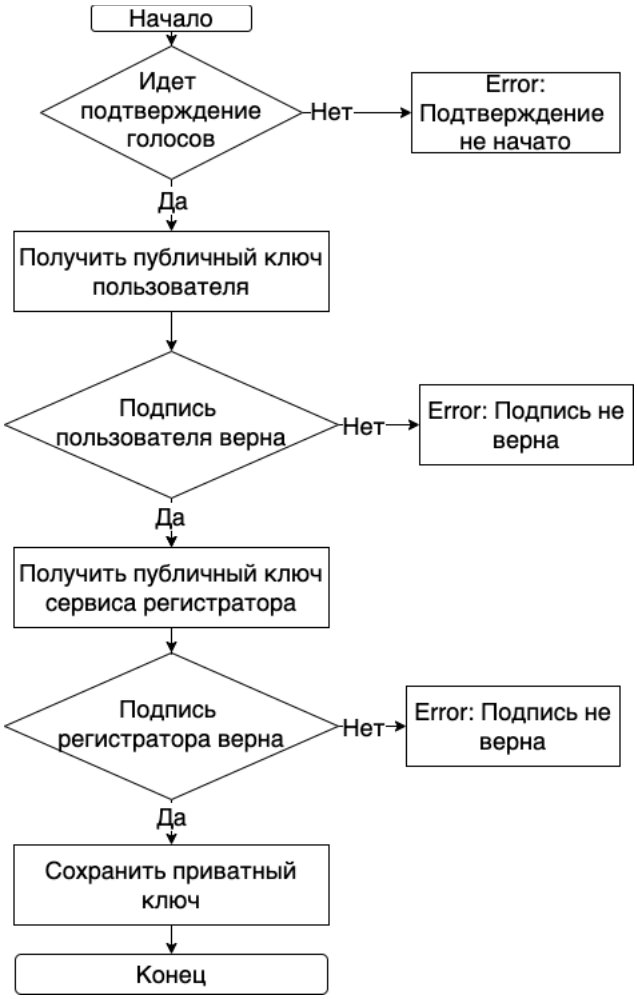



По окончанию голосования становится доступная точка входа get_results, при запросе на нее происходит построчное получение данных из таблицы bulletins. Если для бюллетеня есть ключ дешифрования, то сообщение дешифруется и

суммируется в оперативной памяти, по окончанию выборки, запрос возвращается в виде json. В случае, если бюллетень есть, но ключ для дешифрования пользователь не прислал, такие бюллетени считаются не проголосовавшими ни за одного кандидата, так же, как и пользователи, вообще не прошедшие ни одного этапа голосования. Но такие бюллетени полезны для исследования причин их наличия, например это пользователи специально не подтвердили свой голос, или это технический сбой. В некоторых случаях можно продлевать этап подтверждения голосов, и дополучить недостающие ключи

Алгоритм подтверждения голоса для передачи секретного ключа отображен на рисунке 3.8

Рисунок 3.8 – Блок схема подтверждения голоса сервисом учета голосов



В процессе голосования через промежутки времени указанные в config база данных реплицируется. И может быть получена по запросу на точку входа get_replicas. Копия не шифруется дополнительно, ведь сами бюллетени в ней зашифрованы и узнать содержимое невозможно. По окончанию этапа подтверждения голосов можно отдельно запросить ключи дешифрования, чтобы дешифровать копии. Копии будут запрашивать и хранить наблюдатели по ходу голосования, а также все созданные копии остаются на сервисе учета голосов.

Таймеры запуска этапов голосования так же, как и для сервиса регистратора прописываются в config.

Создание копий базы данных для передачи их наблюдателям реализуем с помощью встроенного в PostgreSQL инструмента pg_dump. Регулярность выполнения реализуем с помощью crontab. Копии сохраняются на сервере и получить копии возможно выполнив get запрос по пути указанному в pgsql_dump.sh

Полный листинг исходного кода находится в приложении Б.

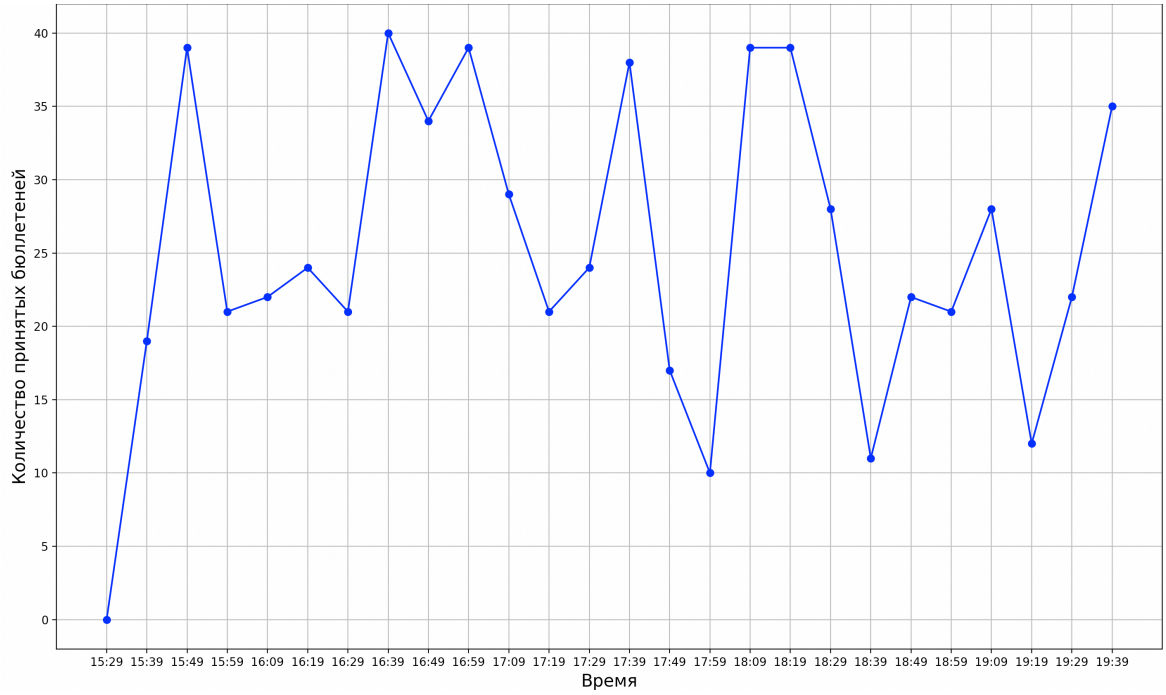
### 3.4 Разработка модуля аудита

Первой частью аудита голосования является подсчет наблюдателями результатов голосования вручную и проверка, что бюллетени не подменялись. Для этого уже реализована часть функционала на сервисе учета голосов: репликация базы данных, выдача на устройства наблюдателей копий базы данных, выдача ключей для дешифровки по окончанию голосования. Наблюдатели сравнивают копии между собой, а также по окончанию голосования дешифрую бюллетени.

Так как реплики приходят во время голосования, хоть они и зашифрованы, на экраны наблюдателей можно вывести данные о количестве проголосовавших. Для этого разработаем ПО, которое при получении новой копии считает количество бюллетеней в этой копии, вычитает количество в прошлой копии и устанавливает точку на линейном графике. Так можно наблюдать за тем сколько людей проголосовало в промежутке времени, сколько осталось и во время голосования

заметить резкие скачки голосования или наоборот полное их отсутствие, как индикацию, что что-то пошло не так. Пример графика изображен на рисунке 3.9

Рисунок 3.9 – Линейный график количества голосовавших по времени




Репликация базы данных происходит и на сервисе регистраторе, получая эти данные можно узнать сколько пользователей получило подписи и сравнивать это количество с количеством принятых бюллетеней. Количество принятых бюллетеней не должно превышать количество подписей.

Так же разработаем ПО для автоматического сравнения последних копий базы данных. ПО сравнивает хеш суммы голосов из предпоследней копии и последней копии. В случае если в новой копии старый голос был изменен или отсутствует ПО сразу же сигнализирует об этом. Новые сообщения, которых не было в предыдущей копии, проверяются на предмет подлинности подписей сервисов регистратора и учета голосов. Это позволит быстро среагировать и предпринять какие-то меры по обнаружению злоумышленника и/или ошибки в работе сервиса. Блок схемы алгоритмов работы ПО изображены на рисунках 3.10 – 3.11

Рисунок 3.10 – Блок схема проверки целостности бюллетеней

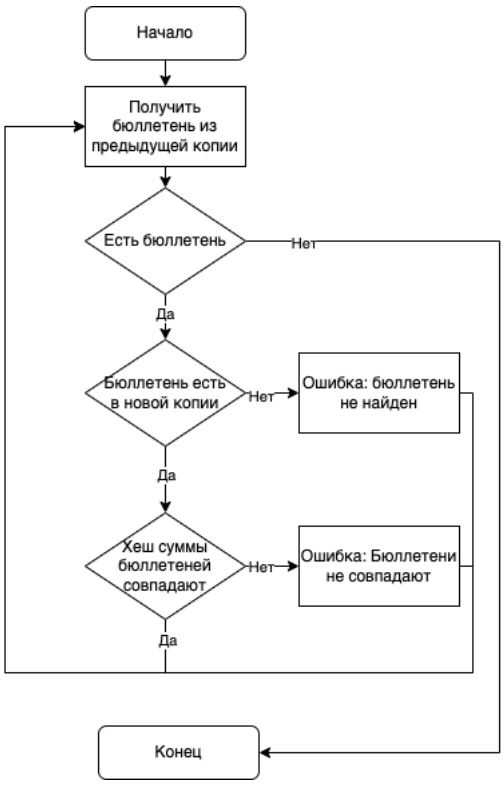
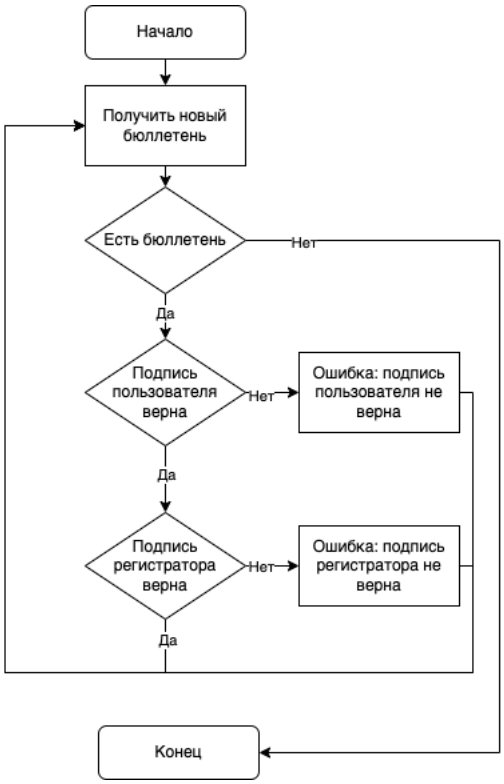
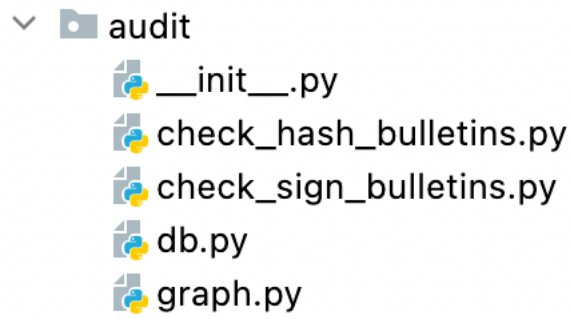



Рисунок 3.11 – Блок схема проверки подписей на бюллетенях



Так же с сервисов регистратора и учета голосов транслируются логи на устройства технических специалистов – наблюдателей, для анализа корректности работы сервисов в течении голосования. Структура модуля аудита отображена на рисунке 3.12

Рисунок 3.12 – Структура модуля аудита




Исходный код модуля аудита представлен в приложении В.

## 3.5 Разработка модуля клиента

В соответствии со входными данными в качестве клиентского приложения будет разработано приложение для компьютера с помощью библиотеки Tkinter.

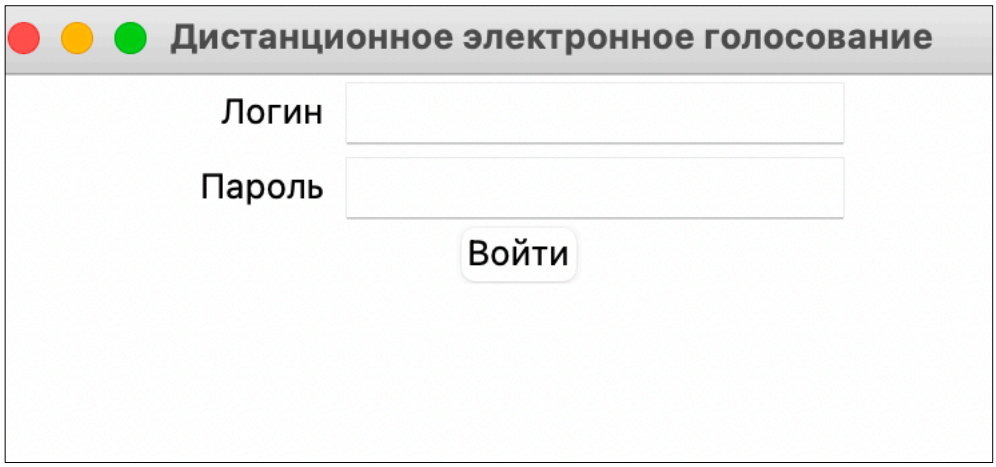
Для клиента понадобится сследующие экраны:

- авторизация,
- текущий статус голосования,
- голосование.

Начнем с экрана авторизации. В рамках ВКР будем пользоваться, нами же разработанным модулем авторизации. Для авторизации пользователя будем использовать связку логин-пароль. Для реализации нам понадобится 2 поля для ввода, 2 текстовых поля и кнопка.

Структура клиентского модуля состоит только из одного файла-скрипта client.py. На рисунке 3.13 изображен интерфейс авторизации.

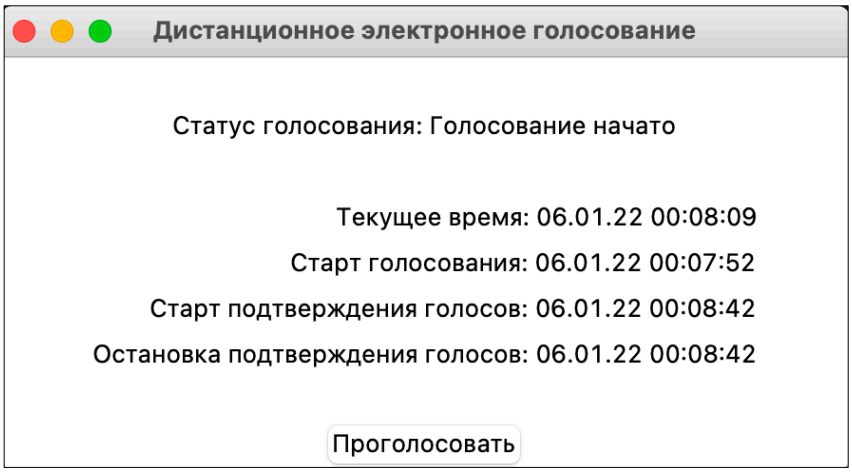
Рисунок 3.13 – Интерфейс авторизации




В случае неудачи появится сообщение с текстом: «Неверный логин или пароль!».

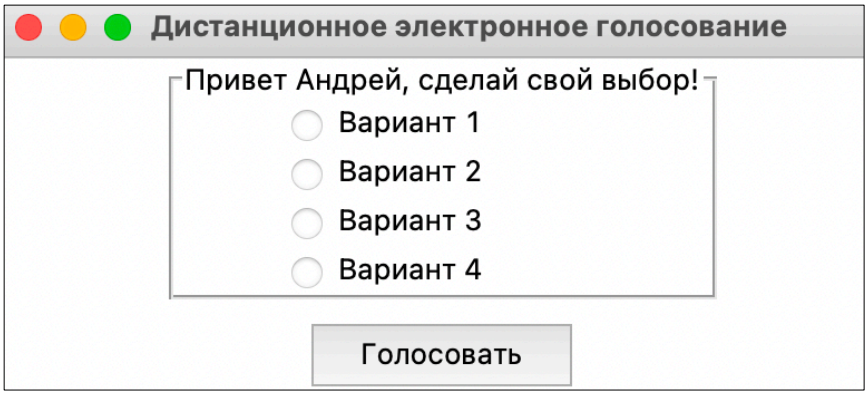
После авторизации пользователь попадает на окно со статусом голосования, где отображен текущий статус голосования и время, когда начнется тот или иной этап. Если в текущий момент идет голосование, то будет отображаться кнопка для перехода к экрану голосования. Если в текущий момент идет подтверждение голосов, то будет отображаться кнопка для передачи секретного ключа. В другое же время кнопки - нет, рисунок 3.14

Рисунок 3.14 – Экран со статусом голосования




По нажатию на кнопку голосования отобразится экран, где можно выбрать кандидата, рисунок 3.15.

Рисунок 3.15 – Экран голосования с вариантами

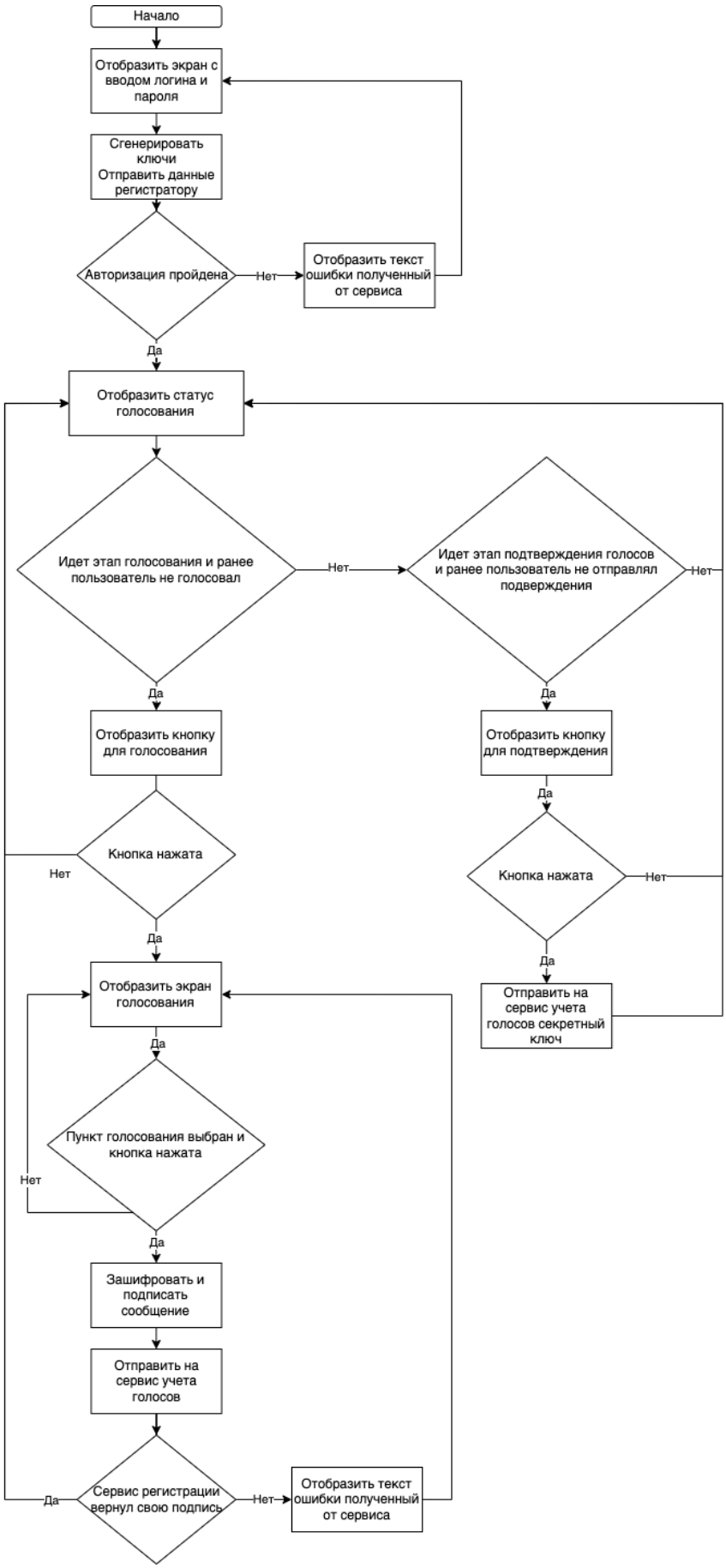



При выборе варианта и нажатии на кнопку «Голосовать» запускается протокол тайного голосования.

В соответствии с протоколом генерируется открытый и закрытый ключ для подписи, ключ для шифрования и ключ для ослепляющего шифрования. Для всех операций будем использовать криптографический алгоритм RSA с размером ключа 4096 бит. Сообщение шифруется, подписывается. На публичный ключ накладывается слой ослепляющего шифрования. Далее на сервис авторизации отправляется логин, пароль и подпись. От сервиса авторизации получаем ключ обратно, уже подписанный сервисом, снимается слой ослепляющего шифрования и сообщение со всеми подписями отправляется на сервис учета голосов. После этого попадаем обратно на экран со статусом, где пользователь ожидает окончания первого этапа голосования и повеления кнопки для отправки секретного ключа. Реализация в виде кнопки, а не автоматическая отправка, так как голосование может длится сколько угодно времени, а пользователь за это время может закрыть программу. По нажатию на кнопку на сервис отправляется ключ дешифрования.

Алгоритм работы клиентского приложения отображен в виде блок-схемы на рисунке 3.16

Рисунок 3.16 – Блок схема работы клиентского приложения



Полный листинг исходного кода клиентского приложения находится в Приложении Г.

## 3.6 Выводы по разделу

В данном разделе была разработана система дистанционного электронного голосования. Удалось решить все задачи, которые были поставлены в техническом задании. Полный листинг кода представлен в приложениях.

# 4 Безопасность жизнедеятельности

## 4.1 Постановка задачи

В данном разделе необходимо рассмотреть следующие вопросы:

- особенности воздействия электронных систем на здоровье пользователей;
- эргономические требования к системам отображения информации;
- режимы труда и отдыха при работе с электронными устройствами;
- экологические проблемы утилизации электронных гаджетов.

## 4.2 Воздействие электронных систем на здоровье пользователей

На пользователя электронных систем может воздействовать ряд опасных и вредных факторов, наиболее значимые из которых следующие:

— Повышенный уровень напряжения в электрических цепях питания и управления ПК, который может привести к электротравме оператора при отсутствии заземления оборудования;

— Излучения от экрана монитора. Как показали результаты многочисленных научных работ с использованием новейшей измерительной техники зарубежного производства, монитор ПК является источником электромагнитного излучения в низкочастотном, высокочастотном и сверхвысокочастотном диапазоне, мягкого рентгеновского излучения от электроннолучевой трубки (ЭЛТ), ультрафиолетового излучения, инфракрасного излучения, электростатического поля

— Не соответствующие нормам параметры микроклимата: повышенная температура из-за постоянного нагрева деталей ПК, пониженная влажность.

— Нарушение норм по аэроионному составу воздуха, особенно в помещениях с разной системой приточно-вытяжной вентиляции и (или) с кондиционерами, при этом концентрация полезных для организма отрицательно заряженных легких

ионов кислорода воздуха (аэроионов) может быть в 10-50 раз ниже нормы, а концентрация вредных положительных ионов значительно превышать норму.

— Пониженный или повышенный уровень освещенности в помещении; не соответствующие санитарным нормам визуальные параметры дисплея. Деятельность оператора предполагает, прежде всего, визуальное восприятие отображаемой на экране монитора информации, поэтому значительной нагрузке подвергается зрительный аппарат работающих с ПК.

— Повышенный уровень шума в системном блоке компьютера.

— Повышенный уровень загазованности воздуха; повышенное содержание в воздухе патогенной особенно зимой при повышенной температуре в помещении, плохом проветривании, пониженной влажности, нарушении аэроионного состава воздуха.

Трудовой кодекс обязывает работодателей обеспечить безопасные условия и охрану труда работников на каждом рабочем месте (ст. 212 ТК РФ)

В соответствии с СанПиНом 2.2.2/2.4.1340–03 выдвигаются следующие требования к помещениям для работы с ПЭВМ:

— В производственных помещениях, в которых работа с использованием ПЭВМ является основной (диспетчерские, операторские, расчетные, кабины и посты управления, залы вычислительной техники и др.) и связана с нервно-эмоциональным напряжением, должны обеспечиваться оптимальные параметры микроклимата для категории работ 1а и 1б в соответствии с действующими санитарно-эпидемиологическими нормативами микроклимата производственных помещений. На других рабочих местах следует поддерживать параметры микроклимата на допустимом уровне, соответствующем требованиям указанных выше нормативов.

— В помещениях всех типов образовательных и культурно-развлекательных учреждений для детей и подростков, где расположены ПЭВМ, должны обеспечиваться оптимальные параметры микроклимата, указанные в приложении 2 СанПиН.

— В помещениях, оборудованных ПЭВМ, проводится ежедневная влажная уборка и систематическое проветривание после каждого часа работы на ПЭВМ.

— Уровни положительных и отрицательных аэроионов в воздухе помещений, где расположены ПЭВМ, должны соответствовать действующим санитарно-эпидемиологическим нормативам.

— Содержание вредных химических веществ в воздухе производственных помещений, в которых работа с использованием ПЭВМ является вспомогательной, не должно превышать предельно допустимых концентраций вредных веществ в воздухе рабочей зоны в соответствии с действующими гигиеническими нормативами.

— Содержание вредных химических веществ в производственных помещениях, в которых работа с использованием ПЭВМ является основной (диспетчерские, операторские, расчетные, кабины и посты управления, залы вычислительной техники и др.), не должно превышать предельно допустимых концентраций загрязняющих веществ в атмосферном воздухе населенных мест в соответствии с действующими гигиеническими нормативами.

В таблице 4.1 приведены временные допустимые уровни ЭМП, создаваемых ПЭВМ на рабочих местах, а в таблице 4.2 – визуальные параметры ВДТ, контролируемые на рабочих местах.

Таблица 4.1 – Временные допустимые уровни ЭМП, создаваемых ПЭВМ на рабочих местах

| Наименование параметров | | ВДУ |
|---|---|---|
| Напряженность электрического поля | в диапазоне частот 5 Гц - 2 кГц | 25 В/м |
| | в диапазоне частот 2 кГц - 400 кГц | 2,5 В/м |
| Плотность магнитного потока | в диапазоне частот 5 Гц - 2 кГц | 250 нТл |
| | в диапазоне частот 2 кГц - 400 кГц | 25 нТл |
| Напряженность электростатического поля | | 15 кВ/м |

Продолжение таблицы 4.1

| Параметры | Допустимые значения |
|---|---|
| Яркость белого поля | Не менее 35 кд/кв. м |
| Неравномерность яркости рабочего поля | Не более +/- 20% |
| Контрастность (для монохромного режима) | Не менее 3:1 |
| Временная нестабильность изображения (мелькания) | Не должна фиксироваться |
| Пространственная нестабильность изображения (дрожание) | Не более 2 x 1E(-4L), где L – проектное расстояние наблюдения, мм |

4.3 Эргономические требования к системам отображения информации

Эргономические требования описаны в ГОСТ Р 50948-2001.

При необходимости распознавания или идентификации цветовых параметров прикладная программа должна предлагать устанавливаемый по умолчанию набор цветов, который соответствует требованиям настоящего стандарта. Если цвет может быть изменен пользователем, то должна быть предусмотрена возможность восстановления назначенного по умолчанию набора цветов.

При необходимости точной идентификации цвета в рядах буквенно-цифровых знаков и в полях ввода данных высота символа должна быть не менее 20' при проектном расстоянии наблюдения.

При необходимости точной идентификации цвета обособленного изображения (например, знака или символа) угловой размер изображения должен быть не менее 30' при проектном расстоянии наблюдения (предпочтительно - 40').

Следует избегать применения насыщенного синего цвета для изображений, имеющих угловой размер менее 2°.

Для чтения текстов, буквенно-цифровых знаков и символов при отрицательной полярности изображения не следует применять синий и красный цвета спектра на темном фоне и красный цвет спектра на синем фоне.

Для чтения текстов, буквенно-цифровых знаков и символов при положительной полярности изображения не следует применять синий цвет спектра на красном фоне.

Насыщенные крайние цвета видимого спектра приводят к нежелательным эффектам глубины изображаемого пространства и не должны применяться для изображений, которые требуют непрерывного просмотра или чтения.

Для точного распознавания и идентификации цветов должны применяться цветное изображение переднего плана на ахроматическом фоне или ахроматическое изображение переднего плана на цветном фоне.

Число цветов, одновременно отображаемых на экране дисплея, должно быть минимальным. Для точной идентификации цвета каждый заданный по умолчанию набор цветов должен включать не более 11 цветов.

При необходимости проведения быстрого поиска, основанного на распознавании цветов, следует применять не более 6 различных цветов.

При необходимости вызова параметров цвета из памяти ЭВМ следует применять не более 6 различных цветов

Яркость знака должна быть не менее 35 кд/м для дисплеев на ЭЛТ и не менее 20 кд/м для плоских дискретных экранов.

Неравномерность яркости рабочего поля экрана должна быть не более 20%.

Неравномерность яркости элементов знака должна быть не более 20%.

Яркостный контраст изображения должен быть не менее 3:1 (для плоских дискретных экранов при угле наблюдения от минус 40° до плюс 40°). Яркостный контраст внутри знака и между знаками должен быть не менее 3:1.

Ширина контура знака должна быть от 0,25 до 0,5 мм.

Степень несведения цветов в любом месте многоцветного экрана для дисплеев на ЭЛТ должна быть не более 3,4' при проектном расстоянии наблюдения.

Изменение размеров однотипных знаков по рабочему полю должно быть в пределах ±5% высоты знака.

Максимальная разность длин строк текста на рабочем поле должна быть не более 2% средней длины строки.

Максимальная разность длин столбцов текста на рабочем поле должна быть не более 2% средней длины столбца.

Отклонение формы рабочего поля от прямоугольника определяют по следующим формулам:

по вертикали

$$\Delta H = 2\frac{H_1 - H_2}{H_1 + H_2} \leq 0{,}02 \quad (4.1)$$

по горизонтали

$$\Delta B = 2\frac{B_1 - B_2}{B_1 + B_2} \leq 0{,}02 \quad (4.2)$$

по диагонали

$$\Delta D = 2\frac{D_1 - D_2}{D_1 + D_2} \leq 0{,}04\frac{H_1 + H_2}{B_1 + B_2} \quad (4.3)$$

где $H_1$, $H_2$, $B_1$, $B_2$, $D_1$ и $D_2$ - значения длин крайнего левого и крайнего правого столбца, верхней, нижней строки и диагоналей на рабочем поле соответственно, мм.

Временная нестабильность изображения (мелькания) для дисплеев на ЭЛТ и на плоских дискретных экранах не должна быть зафиксирована. Для дисплеев на ЭЛТ частота обновления изображения должна быть не менее 75 Гц при всех режимах разложения, гарантируемых нормативной документацией на конкретный тип дисплея и не менее 60 Гц для дисплеев на плоских дискретных экранах.

Амплитуда смещения изображения (пространственная нестабильность изображения - дрожание) должна быть не более 2·10, где - проектное расстояние наблюдения, мм.

Методы контроля эргономических параметров и параметров безопасности описаны в ГОСТ Р 50949.

4.4 Режимы труда и отдыха при работе с электронными устройствами

В течении рабочего дня согласно трудовому кодексу доступны следующие перерывы:

— обеденный перерыв по ст. 108 ТК РФ;

— специальные перерывы, обусловленные технологией и организацией производства и труда по ст. 109 ТК РФ;

— специальные перерывы для отдыха и обогревания по ст. 109 ТК РФ.

Порядок предоставления перерывов устанавливаются правилами внутреннего трудового распорядка.

В Законе «О санитарно-эпидемиологическом благополучии населения» прописано, что критерии безопасности или безвредности условий работ с источниками физических факторов воздействия на человека, в том числе предельно допустимые уровни воздействия, устанавливаются санитарными правилами (п. 2 ст. 27 Закона от 30.03.99 № 52-ФЗ). В этом законе приписаны требования к организации работы за персональными электронно-вычислительными машинами. СанПиН 2.2.2/2.4.1340-03 действовал до 01.01.2021 г.

В нем существовало определение суммарного времени регламентированных перерывов, зависит оно от и уровня нагрузки за рабочую смену, а также от категории трудовой деятельности. При 8-часовой рабочей смене суммарное время перерывов составляет от 50 до 90 минут. При 12-часово от 80 до 140 минут. Если человек в течение 8-часового рабочего дня работает за компьютером 50% рабочего времени (то есть до 4 часов), то суммарные перерывы для отдыха от ПЭВМ должны составлять 70 минут.

То есть необходимо чередовать работу с использованием компьютера и без него, делая небольшие перерывы для отдыха. Работодатель в правилах внутреннего трудового распорядка прописывает время начала и продолжительность каждого перерыва для различных категорий работников сам. Находиться на рабочем месте во время таких перерывов необязательно (ст. 106, 107 ТК РФ).

При работе за компьютером ночью (с 22 до 6 часов) продолжительность регламентированных перерывов следует увеличить на 30% (п. 1.6 Приложения № 7 к СанПиН 2.2.2/2.4.1340-03).

Также время работы за компьютером регулировал такой документ, как Типовая инструкция по охране труда при работе на персональном компьютере (ТОИ Р-45-084-01, утв. Приказом Минсвязи РФ от 02.07.2001 N 162). В ней сказано, что время непрерывной работы за компьютером без регламентированного перерыва не может превышать 2 часов (п. 3.2 ТОИ Р-45-084-01).

Эта инструкция с 01.01.2021 г. утратила силу.

То есть с 2021 г. вопрос установления перерывов во время работы за компьютеры нормативно не урегулирован. Работодатель может самостоятельно установить порядок предоставления перерывов в работе за компьютером для отдыха в правилах внутреннего трудового распорядка. Важно помнить, что указанные перерывы включаются в рабочее время. То есть они не продлевают продолжительность рабочего дня сотрудника. Во время этих перерывов работник не должен выполнять другую работу. Перерыв предоставляется ему для отдыха (Письмо Минтруда от 14.06.2017 № 14-2/ООГ-4765).

Кроме того, важно помнить, что перерывы в работе для отдыха от компьютера нужно предоставлять отдельно от перерыва на обед (ст. 108, 109 ТК РФ).

### 4.5 Экологические проблемы утилизации электронных гаджетов.

Устаревшие персональные компьютеры или их элементы должны быть правильно утилизированы в целях предотвращения вредного воздействия отходов производства и потребления на здоровье человека и окружающую среду, а также вовлечения таких отходов в хозяйственный оборот в качестве дополнительных источников сырья. За несоблюдение законодательства России по утилизации офисной техники на организацию могут быть наложены штрафные санкции. Выбрасывание компьютерной техники ведет к загрязнению окружающей среды. Персональный

компьютер включает в свой состав как органические составляющие (пластик различных видов, материалы на основе поливинилхлорида, фенол формальдегида), так и почти полный набор металлов, в том числе и драгоценных. В связи с этим организации требуется документально контролировать оборот средств компьютерной техники от поступления до выбытия. Согласно Приказу ГТК РФ от 19.11.2002 N 1224 «О порядке учета и хранения изделий и материалов, изготовленных с применением драгоценных металлов и драгоценных камней», организация вправе:

– самостоятельно обрабатывать (перерабатывать) собранный лом, содержащий драгоценные металлы;

– реализовывать лом, содержащий драгоценные металлы;

– передавать на давальческой основе аффинажным организациям или организациям, осуществляющим деятельность по заготовке лома и отходов, первичной обработке и переработке, для дальнейшего производства и аффинажа.

Процесс утилизации компьютерной техники включает следующие пункты:

– создание внутренней комиссии в организации, которая решит, что нужно списать;

– составление экспертного заключения и подтверждение невозможности дальше пользоваться компьютерным оборудованием;

– осуществление списания компьютерной техники, которое будет отражено в бухгалтерском учете;

– утилизация мусора на лицензированном предприятии и получение документального подтверждения о проведенных действиях (акт выполненной работы, приема-передачи).

– утилизация персональных компьютеров имеет определенные сложности в реализации, но это необходимый этап в поддержании экологической ситуации. [9]

## 4.6 Вывод

В данном разделе были описаны особенности воздействия электронных систем на здоровье пользователей, выдвинуты эргономические требования к системам отображения информации в соответствии с нормативными документами. Выяснили, что в данный момент режимы труда и отдыха при работе с электронными устройствами нормативно не урегулирован. Проанализировали экологические проблемы утилизации электронных гаджетов.

# 5 Технико-экономическое обоснование работы

## 5.1 Постановка задачи

Целью выпускной квалификационной работы являлась разработка веб- приложения для защищенного электронного голосования. Веб-приложение является программным кодом, который, согласно ст. 1259 ГК РФ, относится к объектам авторских прав, таким образом, является интеллектуальной собственностью.

В данном разделе будут рассмотрены следующие вопросы:

- расчет трудоемкости и длительности работ;
- расчет себестоимости и цены программного продукта;
- эффект от разработки программного продукта;
- конкурентоспособность продукта.

## 5.2 Расчет трудоемкости и длительности работ

В первую очередь необходимо составить план по разработке программного продукта, который представлен в таблице 5.1.

Таблица 5.1 – План разработки программного продукта

| Наименование этапов | Виды работ | Исполнитель | Количество исполнителей |
|---|---|---|---|
| Анализ предметной области | Определение объекта разработки | Студент | 1 |
| | Анализ основных угроз и уязвимостей | Студент | 1 |
| | Разработка модели нарушителя информационной безопасности | Студент | 1 |
| Проектирование | Проработка концепции | Студент | 1 |
| | Выбор протокола голосования | Студент | 1 |
| | Планирование архитектуры приложения | Студент | 1 |
| Разработка | Разработка сервера авторизации | Студент | 1 |
| | Разработка сервера учета голосов | Студент | 1 |
| | Разработка системы аудита | Студент | 1 |
| Тестирование | Тестирование работоспособности | Студент | 1 |
| | Тестирование защищенности | Студент | 1 |
| Внедрение | Улучшение, оптимизация, устранение ошибок | Студент | 1 |

Далее требуется рассчитать трудоемкость и длительность работ. Поскольку трудоемкость этапов и видов работ носит вероятностный характер, то предпочтительным будет использование метода экспертных оценок.

В этом методе для каждого этапа требуется экспертным путем определить три оценки трудоемкости, в днях:

Далее для каждого этапа определены три величины:

– наиболее вероятная величина затрат, $m_i$;

– наименее возможная величина затрат, $a_i$;

– наиболее возможная величина затрат, $b_i$.

На основании экспертных оценок средняя величина для $m_i$, $a_i$,и $b_i$ определяется по формуле (5.1):

$$\overline{\mathrm{T}} = \frac{3\mathrm{T}_{\text{рук}}+2\mathrm{T}_{\text{авт}}}{5}, \qquad (5.1)$$

где $\overline{T}$ – среднее время, полученное на основании экспертных оценок;

$T_{рук}$ – оценка затрат времени, данная руководителем;

$T_{авт}$ – оценка затрат времени, данная автором проекта.

Результаты расчета средней оценки затрат времени на разработку программного продукта приведены в таблице 5.2 (оценка производится в днях).

Таблица 5.2 – Время, затраченное на разработку программного продукта

| Этапы разработки программного продукта | Наименее возможная величина затрат ($a_i$), дни | | | Наиболее вероятная величина затрат ($m_i$), дни | | | Наиболее возможная величина затрат ($b_i$), дни | | |
|---|---|---|---|---|---|---|---|---|---|
| | $T_{авт}$ | $T_{рук}$ | $\overline{T}$ | $T_{авт}$ | $T_{рук}$ | $\overline{T}$ | $T_{авт}$ | $T_{рук}$ | $\overline{T}$ |
| 1Анализ предметной области | 2 | 2 | 2 | 3 | 4 | 3,6 | 5 | 6 | 5,6 |
| 2 Проектирование | 2 | 3 | 2,6 | 3 | 5 | 4,2 | 4 | 6 | 5,2 |
| 3 Разработка | 4 | 5 | 4,6 | 5 | 6 | 5,6 | 7 | 7 | 7 |
| 4 Тестирование | 1 | 1 | 1 | 2 | 3 | 2,6 | 4 | 5 | 4,5 |
| 5 Внедрение | 2 | 3 | 2,6 | 3 | 4 | 3,6 | 5 | 5 | 5 |

На основе средних оценок рассчитываются отклонение по каждому этапу разработки программного продукта и математическое ожидание. Формула расчета математического ожидания для i-го этапа:

$$MO_i = \frac{a_i + 4m_i + b_i}{6}, \qquad (5.2)$$

где $MO_i$ – математическое ожидание для i-го этапа;

$a_i, m_i, b_i$ – средние значения.

Стандартное отклонение для каждого этапа разработки определяется по формуле:

$$G_i = \frac{b_i - a_i}{6}, \qquad (5.3)$$

где $G_i$ – стандартное отклонение по i-му этапу.

Зная математическое ожидание по каждому этапу, рассчитывается общая величина математического ожидания в целом по программному средству:

$$MO = \sum MO_i, \qquad (5.4)$$

где MO – общая величина математического ожидания.

Стандартное отклонение G в целом по программному средству рассчитывается по следующей формуле:

$$G = \sqrt{\sum G_i^2}, \qquad (5.5)$$

где G –стандартное отклонение;

$G_i$ – стандартное отклонение по i-му этапу.

На основе расчетов математического ожидания (5.2) и стандартного отклонения (5.3) рассчитывается коэффициент вариации – коэффициент согласованности мнения экспертов. Коэффициент вариации рассчитывается по формуле:

$$v_i = \frac{G_i}{MO_i}, \qquad (5.6)$$

где $y_i$ – коэффициент вариации по i-му этапу.

Теперь можно произвести расчеты на основе таблицы 5.3 и формул (5.2 – 5.6) и свести эти расчеты в таблицу 5.3.

Таблица 5.3 – Затраты на разработку программного продукта

| Этапы разработки программного продукта | Средняя величина затрат по этапам, дни | | | Матем. ожидание ($MO_i$, дни) | Станд. Отклонение ($G_i$, дни) | Коэффициент вариации ($v_i$) |
|---|---|---|---|---|---|---|
| | Наименее возможная величина затрат ($a_i$, дни) | Наиболее вероятная величина затрат ($m_i$, дни) | Наиболее возможная величина затрат ($b_i$, дни) | | | |
| 1 Анализ предметной области | 2 | 3,6 | 5,6 | 3,67 | 0,6 | 0,16 |
| 2 Проектирование | 2,6 | 4,2 | 5,2 | 4,1 | 0,43 | 0,1 |
| 3 Разработка | 4,9 | 5,6 | 7 | 5,72 | 0,35 | 0,06 |
| 4 Тестирование | 1 | 2,6 | 4,5 | 2,65 | 0,58 | 0,22 |
| 5 Внедрение | 2,6 | 3,6 | 5 | 3,67 | 0,4 | 0,11 |
| Итого | 13,1 | 19,6 | 27,3 | 19,81 | 1,08 | 0,13 |

Коэффициент вариации равен 0,13 и не превосходит **0,33**. Поэтому мнения экспертов считают согласованными.

## 5.3 Расчет себестоимости программного продукта

Себестоимость программного продукта – это все виды затрат, понесённые при разработке продукта. Себестоимость включает в себя:

- затраты на материалы;
- трудовые затраты;
- амортизацию основных средств;

– прочие (накладные расходы, затраты сторонних организаций и т.д.).

Чтобы определить себестоимость разработки программного продукта применяется метод экспертных оценок. Данный метод заключается в следующем: оценка затрат производится несколькими экспертами на основании собственного опыта и знаний. В данном случае в качестве экспертов выступают автор проекта и руководитель. Использование данного метода оправдано, так как процесс написания программы является творческим и поэтому сложно ввести нормативы для оценки затрат.

Себестоимость программного продукта определяется по формуле

$$C = \frac{З}{m} \cdot k \cdot k_{ТЕР} \cdot k_{ПР} \cdot t1 + t2 + t3 + t4 \cdot 1 + kn + 8 \cdot t3 \cdot См + 8 \cdot t4 \cdot Си, \quad (5.7)$$

где З - среднемесячная заработная плата разработчика программы = 40000;

$k_{ТЕР}$ - территориальный коэффициент, $k_{ТЕР}$ = 1,2 (для НСО);

$k_{ПР}$ - коэффициент премии $k_{ПР}$ = 1;

$k$ - коэффициент, учитывающий страховые взносы (фонды пенсионного, социального и медицинского страхования), $k = 1,3$

$m$ - количество рабочих дней в месяце, $m$ = 22;

$K_Н$ = - коэффициент, учитывающий накладные расходы (отопление, освещение, уборка и т. д.), Кн = 0,4;

$t_1$ - время, затраченное разработчиком на разработку требований к программе, т.е. подготовительное время, которое необходимо потратить, чтобы преступить к написанию программы и отладки программы, чел./дни;

$t_2$ – сборка устройства, составление алгоритма в программе, время, затраченное на написание и отладку программы, чел./дни;

$t_3$ – время, затраченное на разработку программы с использованием машинного времени, чел./дни;

$t_4$ – время работы в сети интернет, дни;

$С_и$ - стоимость 1 часа работы в сети интернет, руб. Стоимость работы в сети Интернет оценивается по входящему трафику (через абонентскую плату или через количество мегабайт информации).

$C_т$. - стоимость одного часа машинного времени.

*8* – количество рабочих часов в день.

Для расчета стоимости одного часа машинного времени необходимо определить затраты на эксплуатацию ПК за год.

$$C_м = \frac{З_{эл} + З_а + З_{компл} + З_{пр}}{T_{общ}}, \qquad (5.8)$$

где $C_м$ – стоимость одного часа машинного времени;

$T_{общ}$ – общее время работы компьютера в год;

$З_{эл}$ – затраты на электроэнергию за год работы;

$З_а$ – амортизационные отчисления;

$З_{компл}$ – затраты на комплектующие материалы;

$З_{пр}$ – прочие расходы.

Общее время работы компьютера за год составляет:

$Т_{общ} = 22 * 12 * 8 = 2112$ часов.

Затраты на электроэнергию за год работы (на данный момент тариф $С_{эл}$ составляет 2,49 руб. за кВт-ч):

$$З_{эл} = T_{общ} * C_{эл} * P \qquad (5.9)$$

где Р - потребляемая мощность компьютера по паспортным данным в час, в среднем Р составляет: 450 Вт*ч.

По формуле (5.9) затраты на электроэнергию за год работы составляют:

$$З_{эл} = 2112*2,49*0,45 = 2366,5 \text{ руб.}$$

Амортизационные отчисления в год определяются как процент отчисления на амортизацию от первоначальной стоимости основных производственных фондов. Процент отчисления на амортизацию ($П_р$) согласно статье 258 НК РФ составляет 34-50% от первоначальной стоимости ПК (компьютер относится ко второй

группе имущества со сроком полезного использования свыше 2 лет до 3 лет включительно).

$$З_а = С * П_р \qquad (5.10)$$

где С – стоимость ноутбука, руб.;

Пр – процент отчисления на амортизацию, Пр = 40%.

Стоимость ПК составляет 60 тыс.руб., следовательно, стоимость будет списана единоразово на «Коммерческие и управленческие расходы».

Затраты на ПК составят:

$$Зa = 60000 * 0{,}4 = 24000 \text{ (руб.)}$$

Примем затраты на комплектующие материалы:

$$З_{компл} = 3000 \text{ руб.}$$

Прочие расходы составляют 5% от общей суммы затрат:

$$З_{пр} = \frac{0{,}05 * (З_{эл} + З_а + З_{компл})}{0{,}95} \qquad (5.11)$$

По формуле (5.11) прочие расходы равны:

$$З_{пр} = \frac{0{,}05*(2366{,}5+24000+3000)}{0{,}95} = 1545{,}61 \text{ руб.}$$

Согласно формуле (5.8), стоимость одного часа машинного времени рассчитывается.

$$С_м = \frac{12366{,}5+24000+3000+1545{,}61}{2112} = 19{,}37 \text{ руб.}$$

Стоимость 1 часа работы в сети интернет определим через затраты на абонентскую плату. В среднем, тариф на услуги интернет составляет 600 руб. в месяц, следовательно:

$$С_и = \frac{600}{30} = 20 \text{ руб.}$$

Заключительным этапом является распределением ранее рассчитанной трудоемкости (таблица 3.4) по 4 направлениям:

– $t_1$ включает первые три этапа: анализ предметной области и проектирование:

$t_1$ = 3,67+4,1=7,77 дни

– $t_2$ включает этапы: разработка, тестирование и внедрение:

$t_2$ = 5,72+2,65+3,67=12,04 дни

– $t_3$ включает время работы ПК для разработки программы:

$t_3$ = 20 дни

– $t_4$ включает время использования интернета для разработки программы.

$t_4$ = 20 дни

Таким образом, мы имеем все необходимые данные для расчета себестоимости программного продукта:

$$С = \frac{40000}{22} \cdot 1{,}3 \cdot 1{,}2 \cdot 1 \cdot (7{,}77 + 12{,}04) * (1 + 0{,}4) + 8 * 20 \cdot 19{,}37 + 8 \cdot 20 \cdot 20$$

= 84962,9 руб.

5.4 Расчет цены программного продукта

В случае, если программный продукт будет реализован на рынке, следует рассчитать цену по формуле (5.12):

$$Ц = С \cdot \left(1 + \frac{Р}{100}\right), \qquad (5.12)$$

где *С* – себестоимость разработки программы (согласно формулы 1 находится), руб.;

*Р* – рентабельность, руб.

Определим цену программного продукта, при условии, что значение рентабельности равно 20%:

$$Ц = 84962{,}9 \cdot \left(1 + \frac{20}{100}\right) = 101955{,}48 \text{ руб.}$$

Цена с учетом налога на добавленную стоимость находится по формуле (5.13):

$$Ц_{НДС} = Ц \cdot К_{НДС}, \quad (5.13)$$

где Ц – цена программного продукта;

$К_{НДС}$ – коэффициент, учитывающий ставку налога на добавленную стоимость (НДС), $К_{НДС} = 1{,}20$.

Цена с учетом налога на добавленную стоимость составит:

$Ц_{ндс}$ = 101955,48*1,20 = 122346,56 руб.

5.5 Определение эффекта от разработки программного продукта

Эффект характеризуется экономией рабочего времени при использовании программного продукта. При использовании данной программы автоматизируются стандартные и повседневные операции, что позволяет экономить денежные средства и сокращать время для решения повседневных задач.

Использование электронной системы для голосования даст эффект, как для конечного пользователя, так и для организатора голосования.

Рассмотрим положительные и отрицательные стороны. Для клиентов эффектом будет экономия времени. Появляется возможность проголосовать без непосредственного выезда на место проведения. При выполнении голосования в бумажном виде. Необходимо подготовить место голосования, бюллетени, выдать бюллетени подсчитать их. С авторской программой большинство действий полностью автоматизировано и не требует участия человека.

Результаты расчета о временных затратах на выполнение алгоритма работы голосования до внедрения автоматизированного программного средства приведены в таблице 5.4.

Таблица 5.4 - Оценка затрат времени на выполнение алгоритма работы голосования до внедрения автоматизированного программного средства

| Шаг | Описание процессов | Время, час. |
|---|---|---|
| 1 | Составление списка голосующих | 1 |
| 2 | Организация места проведения | 1 |
| 3 | Выдача бюллетеней для голосования | 0,5 |
| 4 | Подсчет результатов голосования | 1 |
| 5 | Уведомление о результатах голосования | 0,5 |
| | Итого | 4 |

Результаты расчета о затратах времени на выполнение алгоритма после внедрения системы дистанционного электронного голосования приведены в таблице 5.4.

Таблица 5.5 - Оценка затрат времени на выполнение алгоритма работы голосования после внедрения автоматизированного программного средства

| Шаг | Описание | Время, час. |
|---|---|---|
| 1 | Составление списка голосующих | 0,5 |
| 2 | Организация места проведения | 0 |
| 3 | Выдача бюллетеня для голосования | 0 |
| 4 | Подсчет результатов голосования | 0 |
| 5 | Уведомление о результатах голосования | 0 |
| | Итого | 0,5 |

Экономия времени при проведении одного голосования

$$\Delta T_1 = 3{,}5 \text{ч}.$$

Определим общую экономию времени:

$$\Delta T_{общ} = \Delta T_1 \cdot n, \qquad (5.14)$$

где $\Delta T_1$ – экономия времени при проведении одного голосования;

n – среднее количество голосований за день.

Метод наблюдения позволил определить среднее количество голосований за день: 5 ед. Соответственно экономия времени за день составляет:

$$\Delta T = 3{,}5 \cdot 5 = 17{,}5 ч.$$

Общая экономия времени за месяц составляет:

$$\Delta T_{общ} = 17{,}5 * 24 = 420 ч.$$

По формуле (3.2) определим условную экономию численности персонала:

$$\Delta Ч_{усл} = \frac{420 * 12}{1970} \cdot 1{,}08 = 2{,}76,$$

По формуле (3.3) находим годовую экономию по оплате труда с учетом страховых взносов:

$$\Delta Э_{от} = 2{,}76 * 35000 * 12 * 1{,}30 * 1{,}2 = 1808352 \text{ руб.}$$

Таким образом, при использовании разрабатываемого программного продукта, на производстве происходит условная экономия численности персонала, равная 2,76 шт.ед., а также условная экономия денежных средств в размере 1808352 рублей в год. Использование данного программного средства позволяет значительно повысить эффективность проведение голосования.

## 5.6 Оценка конкурентоспособности программного продукта

После расчета себестоимости и цены программного продукта, необходимо проанализировать рынок конкурентов по данному направлению и выявить конкурентные преимущества авторского продукта.

Анализ рыночной ситуации показал, что на рынке имеется 3 аналога авторского приложения.

Аналогами являются программные продукты:

— дистанционное электронное голосование ЦИК РФ;

— E-voting;

— ВТБ регистратор.

С помощью методики анализа потребительских характеристик товаров (услуг) проведем сравнительный анализ авторского приложения с его аналогами и занесем результаты в таблицу 5.5.

В качестве параметров, оказывающих влияние на уровень конкурентоспособности продукции, были выделены следующие:

— доступ к приложению с любого компьютера, имеющего выход в сеть интернет;

— тайна голосования;

— сокрытие результатов до окончания голосования;

— аудит хода голосования;

— данные авторизации и результаты голосования отделены друг от друга;

— Возможность подключения различных способов авторизации;

— голосующий может удостовериться в том, что его голос был учтен верно.

Цену приложения как параметр не используем, потому что голосование от ЦИК РФ является бесплатным для пользователей, и авторское приложение может быть использовано так же и для государственных выборов.

Таблица 5.6 – Сравнительная характеристика аналогов

| № | Параметры сравнения | Программы | | | |
|---|---|---|---|---|---|
| | | Авторское приложение | ДЕГ ЦИК РФ | E-voting | ВТБ регистратор |
| 1 | Доступ к приложению с любого компьютера, имеющего выход в сеть интернет | + | + | + | + |
| 2 | Тайна голосования | + | + | + | + |
| 3 | Сокрытие результатов до окончания голосования | + | + | - | - |
| 4 | Аудит хода голосования | + | + | + | + |
| 5 | Данные авторизации и результаты голосования отделены друг от друга | + | – | – | – |
| 6 | Возможность подключения различных способов авторизации | + | – | – | – |
| 7 | Голосующий может удостовериться в том, что его голос был учтен верно | + | – | – | – |

5.7 Выводы по разделу

В данном разделе определили, что разработка данного программного продукта займет около 20 дней, по себестоимости 84962,9 руб. С учетом налога на добавленную стоимость цена составит 122346,56 руб.

При использовании разрабатываемого программного продукта происходит условная экономия денежных средств в размере 1808352 рублей в год.

Так же выяснили, что продукт конкурентоспособен. Продукт имеет те же параметры, что и у конкурентов, а также обладает параметрами, которых у конкурентов – нет.

В связи с этим делаем вывод, что разработка данного программного продукта является экономически обоснованным.

Заключение

В результате выполнения выпускной квалификационной работы была достигнута поставленная цель и ее задачи.

В первой главе был определен объект разработки, определены требования к ДЭГ, спрогнозированы угрозы и уязвимости разрабатываемой системы и рассмотрены способы их предотвращения. Также была разработана модель потенциального нарушителя информационной безопасности для электронного голосования.

В второй главе были проработаны технические решения для разработки системы дистанционного электронного голосования. Для реализации системы дистанционного электронного голосования был выбран протокол Фудзиока-Окамото-Охта.

В третьей главе была разработана система дистанционного электронного голосования. Система голосования представляет собой сервер регистратор, сервер учета голосов, систему аудита и клиентское приложение. Полный исходный код представлен в приложениях.

В четвертой главе были проработаны вопросы безопасности жизнедеятельности

В пятой главе было выполнено технико-экономическое обоснование и сделан вывод, что разработка данного программного продукта является экономически обоснованным.

Список литературы


1 Доклад на тему: Практика и перспективы использования электронной подписи в ГАС «Выборы» [Электронный ресурс] / cikrf. – URL: http://cikrf.ru/banners/sov_2013/may/04.php (дата обращения: 23.12.2021).

2 Федеральный закон от 12.06.2002 N 67-ФЗ (ред. от 04.06.2021) "Об основных гарантиях избирательных прав и права на участие в референдуме граждан Российской Федерации". Статья 64.

3 Богдан Ю.И. Анализ существующих систем голосования / О.О. Веселая // Харьковский национальный университет радиоэлектроники. – 2011. – С 1-3.

4 Hannu Nurmi, Arto Salomaa. Conducting secret ballot elections in computer networks: Problems and solutions (англ.) // Annals of Operations Research 51 (1994) - С 185-194.

5 Model of The Scheme [Электронный ресурс] / cs.cmu – URL: https://www.cs.cmu.edu/~qihe/paper/e_voting/node2.html (дата обращения: 23.12.2021).

6 Fujioka, Atsushi; Okamoto, Tatsuaki; Ohta, Kazuo. A practical secret voting scheme for large scale elections (англ.) // Lecture Notes in Computer Science. — 1993. — Глава. 718. — С. 244—251.

7 Cranor, Lorrie Faith; Cytron, K. Ron. Sensus: A Security-Conscious Electronic Polling System for the Internet (англ). - IEEE Computer Society Washington, DC, USA.

8 Qi He and Zhongmin Su. A New Practical Secure e-Voting Scheme [Электронный ресурс] / cs.cmu – URL: http://www.cs.cmu.edu/~qihe/paper/e_voting/ (дата обращения: 23.12.2021).

9 Утилизация отходов компьютерной техники и компьютеров [Электронный ресурс] / vtorothody. – URL: https://vtorothody.ru/utilizatsiya/kompyuternoj-tehniki-i-kompyuterov.html (дата обращения: 17.11.2021).

# Приложение А. Сервис регистратор

## start_registrator.py

```
from datetime import datetime
from base64 import b64decode, b64encode
from flask import Flask
from flask import request, jsonify
from registrator.config import Config
from Crypto.PublicKey import RSA
from registrator.helper import auth, save_public_key_user, public_key_by_id

from Crypto.Signature import pkcs1_15
from Crypto.Hash import SHA256



app = Flask(__name__)
app.config.from_object(Config)

PRIVATE_KEY = RSA.generate(4096)
PUBLIC_KEY = PRIVATE_KEY.publickey()


def sign(encrypted_2_message, private):
    hash_encrypted_2_message = SHA256.new(encrypted_2_message)

    signature = pkcs1_15.new(private).sign(hash_encrypted_2_message)
    return signature


def check_sign(encrypted_2_message, public_key, sign):
    """Проверка подписи от пользовательля регистратором"""
    hash_encrypted_message = SHA256.new(encrypted_2_message)
    try:
        pkcs1_15.new(public_key).verify(hash_encrypted_message, sign)
    except:
        return False
    return True


@app.route('/public/<int:id_user>')
def get_public_by_id(id_user):
    return public_key_by_id(id_user)[0]


@app.route('/public')
def get_public():
    return PUBLIC_KEY.export_key()


@app.route('/auth', methods=['POST'])
def auth_route():
    data = request.json
    result = auth(data.get('username'), data.get('password'))
    if result:
        save_public_key_user(result.get('id'), data.get('public_key'))
        return jsonify(dict(result)), 200
    else:
        return {'error_message': 'Неверный логин или пароль'}, 401
```

```
@app.route('/status')
def status():
    return jsonify({
        'start': Config.START_VOTING.strftime('%d.%m.%y %H:%M:%S'),
        'accepting': Config.START_ACCEPTING_VOTE.strftime('%d.%m.%y %H:%M:%S'),
        'stop_voting': Config.STOP_VOTING.strftime('%d.%m.%y %H:%M:%S'),
    }), 200


@app.route('/vote', methods=['POST'])
def vote():

    date_time_now = datetime.now()

    if Config.START_VOTING > date_time_now:
        return {'error_message': 'Голосование еще не начато'}, 200
    elif Config.START_ACCEPTING_VOTE > date_time_now > Config.START_VOTING:
        data = request.json
        sign_user = b64decode(data.get('sign').encode())
        message_user = b64decode(data.get('encrypted_message').encode())
        public_key_user = public_key_by_id(data.get('id'))

        if public_key_user:
            checked = check_sign(message_user, RSA.importKey(public_key_user[0]),
sign_user)
        else:
            return {'error_message': 'Участник не подтвердил возможность голосо-
вать'}, 403

        if checked:
            sign_registrator = sign(message_user, PRIVATE_KEY)
            return jsonify({
                'sign': b64encode(sign_registrator).decode()
            }), 200
        else:
            return {'error_message': 'Бюллетень не подписан участником'}, 403

    elif Config.STOP_VOTING > date_time_now > Config.START_ACCEPTING_VOTE:
        return {'error_message': 'Голосование завершено, идет подтверждение голо-
сов'}, 403
    elif date_time_now > Config.STOP_VOTING:
        return {'error_message': 'Голосование завершено'}, 200
    else:
        return {'error_message': 'Внутренняя ошибка сервера'}, 500


if __name__ == "__main__":
    app.run(host=app.config.get('HOST'), port=app.config.get('PORT'))
```

## config.py

```
import os
from datetime import datetime, timedelta

BASE_DIR = os.path.abspath(os.path.dirname(__file__))


class Config(object):
    SECRET_KEY = '123jp2j1!@E@!ejdasdqo34#$'
    DEBUG = False
    HOST = '0.0.0.0'
    PORT = 13451
    START_VOTING = datetime.now() + timedelta(seconds=10)
    START_ACCEPTING_VOTE = datetime.now() + timedelta(seconds=60)
    STOP_VOTING = datetime.now() + timedelta(seconds=60)
```

## helper.py

```
import uuid
import hashlib
from db import select_one, insert, select

DEFAULT_SALT = 'b6c7130abc3e431b9d0df698d1eea4d5'  # Вторая соль, не хранящаяся в
бд одинаковая для всех паролей


def save_public_key_user(id_user: int, public_key: str) -> bool:
    exist_key = public_key_by_id(id_user)
    if not exist_key:
        sql = f"""
          insert into "public_keys"
      (id_user, key)
      values(
        {id_user}
        , '{public_key}'
      )
        """
        insert(sql)
        return True
    else:
        return False


def public_key_by_id(id_user: int):
    sql = f"""
      select key from "public_keys" where id_user = {id_user}::int
    """
    return select_one(sql)


def auth(login: str, password: str):
    """
    Функция авторизации пользователя
    Args:
        login: логин
        password: пароль

    Returns:
        идентификатор пользователя
```

```
    """
    user = get_user_by_login(login)
    if user and check_password(user['hash_pass'], password):
        return user


def registration(login: str, password: str) -> int:
    """
    Функция регистрации пользователя
    Args:
        login: логин
        password: пароль

    Returns:
        идентификатор пользователя
    """
    hash_pass = hash_password(password)
    sql = f"""
    insert into auth (login, hash_pass) values ('{login}', '{hash_pass}') return-
ing id
    """
    return insert(sql)['id']


def get_user_by_login(login: str):
    """Получение пользователя по логину"""
    sql_query = f"""
    select * from auth where login = '{login}'
    """
    return select_one(sql_query)


def get_salt():
    """Метод возвращает соль"""
    return uuid.uuid4().hex


def hash_password(password: str):
    """Функция хеширования пароля"""
    salt = get_salt()
    return hashlib.sha256(DEFAULT_SALT.encode() + salt.encode() + password.en-
code()).hexdigest() + salt


def check_password(hashed_password, user_password):
    """Проверка пароля и хеша на соответствие"""
    len_salt = len(get_salt())
    password = hashed_password[:-len_salt]
    salt = hashed_password[-len_salt:]
    return password == hashlib.sha256(DEFAULT_SALT.encode() + salt.encode() +
user_password.encode()).hexdigest()
```

db.py

```
import psycopg2
import psycopg2.extras
from pprint import pprint


def connect():
    conn = psycopg2.connect(dbname='registrator', user='raldenprog',
```

```
                                password='asd2ad12@!sda', host='localhost')
    return conn, conn.cursor(cursor_factory=psycopg2.extras.DictCursor)


def select(sql: str):
    pprint(sql)
    conn, cursor = connect()
    cursor.execute(sql)
    return cursor.fetchall()


def select_one(sql: str):
    pprint(sql)
    conn, cursor = connect()
    cursor.execute(sql)
    return cursor.fetchone()


def insert(sql: str):
    pprint(sql)
    conn, cursor = connect()
    cursor.execute(sql)
    conn.commit()
    try:
        return cursor.fetchone()
    except:
        pass
```

start_validator.py

```
from base64 import b64decode, b64encode
from flask import Flask
from flask import request, jsonify
from validator.config_validator import Config
from Crypto.PublicKey import RSA
from validator.helper import insert_message, update_private, check_sign, sign, de-
crypt_messages, get_keys
import requests

app = Flask(__name__)
app.config.from_object(Config)

PRIVATE_KEY = RSA.generate(4096)
PUBLIC_KEY = PRIVATE_KEY.publickey()


@app.route('/vote', methods=['POST'])
def vote():
    data = request.json
    sign_user = b64decode(data.get('sign').encode())
    sign_registrator = b64decode(data.get('sign_registrator').encode())
    message_user = b64decode(data.get('encrypted_message').encode())
    id_user = data.get('id')
    public_key_user_r = requests.get(f'http://0.0.0.0:13451/public/{id_user}')
    public_key_user = public_key_user_r.content.decode()
    public_key_registrator_r = requests.get(f'http://0.0.0.0:13451/public')
    public_key_registrator = public_key_registrator_r.content.decode()

    checked_user_sign = check_sign(message_user, RSA.importKey(public_key_user),
sign_user)
    checked_registrator_sign = check_sign(message_user, RSA.importKey(pub-
lic_key_registrator), sign_registrator)

    if checked_user_sign and checked_registrator_sign:
        message_user_str = data.get('encrypted_message')
        insert_message(id_user, message_user_str)
        sign_validator = sign(message_user, PRIVATE_KEY)
        return jsonify({
            'sign': b64encode(sign_validator).decode()
        })

    else:
        raise Exception('Подписи не верны!')


@app.route('/accept', methods=['POST'])
def accept():
    data = request.json
    id_user = data.get('id')
    private = data.get('private')
    update_private(id_user, private)
    return {}, 200


@app.route('/get_results', methods=['POST'])
def get_results():
```

```
    results = decrypt_messages()
    return jsonify(results), 200


@app.route('/get_keys_for_replicas', methods=['POST'])
def get_keys_for_replicas():
    results = get_keys()
    return jsonify(results), 200


if __name__ == "__main__":
    app.run(host=app.config.get('HOST'), port=app.config.get('PORT'))
```

helper.py

```
from validator.db import insert, select
from Crypto.Cipher import PKCS1_OAEP


from Crypto.Signature import pkcs1_15
from Crypto.Hash import SHA256


def sign(encrypted_2_message, private):
    hash_encrypted_2_message = SHA256.new(encrypted_2_message)

    signature = pkcs1_15.new(private).sign(hash_encrypted_2_message)
    return signature


def check_sign(encrypted_2_message, public_key, sign):
    """Проверка подписи от пользоваетеля регистратором"""
    hash_encrypted_message = SHA256.new(encrypted_2_message)
    try:
        pkcs1_15.new(public_key).verify(hash_encrypted_message, sign)
    except:
        return False
    return True


def insert_message(id_user: int, message: str) -> None:
    sql = f"""
      insert into "bulletins"
  (message, id_user)
  values(
    '{message}', {id_user}
  )
    """
    return insert(sql)


def update_private(id_user: int, private) -> None:
    sql = f"""
      update "bulletins" set private_key = '{private}' where id_user = {id_user}
    """
    return insert(sql)


def decrypt(encrypt_key, message):
    return PKCS1_OAEP.new(encrypt_key).decrypt(message)
```

```
def decrypt_messages():
    sql = f"""
      select * from "bulletins" where private_key is not null
    """
    result_sql = select(sql)
    results = {}
    for row in result_sql:
        decrypted = decrypt(row['messsage'], row['private_key'])
        count = results.get(decrypted)
        results[decrypted] = count
    return results

def get_keys():
    sql = """
    select id, private_key from "bulletins" where private_key is not null
    """
```

db.py

```
import psycopg2
import psycopg2.extras
from pprint import pprint


def connect():
    conn = psycopg2.connect(dbname='validator', user='raldenprog',
                            password='sadw123dw123123', host='localhost')
    return conn, conn.cursor(cursor_factory=psycopg2.extras.DictCursor)


def select(sql: str):
    pprint(sql)
    conn, cursor = connect()
    cursor.execute(sql)
    return cursor.fetchall()


def select_one(sql: str):
    pprint(sql)
    conn, cursor = connect()
    cursor.execute(sql)
    return cursor.fetchone()


def insert(sql: str):
    pprint(sql)
    conn, cursor = connect()
    cursor.execute(sql)
    conn.commit()
    try:
        return cursor.fetchone()
    except:
        pass
```

config_validator.py

```
import os

BASE_DIR = os.path.abspath(os.path.dirname(__file__))
```

```
class Config(object):
    SECRET_KEY = '5756jp2j1!@E@!djfhsakjdh23#$'
    DEBUG = False
    HOST = '0.0.0.0'
    PORT = 13452
```

### pgsql_dump.sh

```
#!/bin/sh

PATH=/etc:/bin:/sbin:/usr/bin:/usr/sbin:/usr/local/bin:/usr/local/sbin

PGPASSWORD=sadw123dw123123
export PGPASSWORD
pathB=/mnt/backup
dbUser=raldenprog
database=validator


find $pathB \( -name "*-1[^5].*" -o -name "*-[023]?.*" \) -ctime +61 -delete
pg_dump -U $dbUser $database | gzip > $pathB/pgsql_$(date "+%Y-%m-%d").sql.gz


unset PGPASSWORD
```

## Приложение В. Модуль аудита

### check_sign_bulletins.py

```
from db import insert, select
from Crypto.Hash import SHA256
import hashlib
from Crypto.Signature import pkcs1_15
import requests
from Crypto.PublicKey import RSA


def check_sign(encrypted_2_message, public_key, sign):
    """Проверка подписи от пользоваетеля регистратором"""
    hash_encrypted_message = SHA256.new(encrypted_2_message)
    try:
        pkcs1_15.new(public_key).verify(hash_encrypted_message, sign)
    except:
        return False
    return True


def check():
    sql = '''
select * from bulletins
    '''
    result_sql = select(sql)

    for row in result_sql:
        id_user = row['id_user']
        message_user = row['message']
        sign_user = row['sign_user']
        sign_registrator = row['sign_registrator']
        public_key_user_r = requests.get(f'http://0.0.0.0:13451/public/{id_user}')
        public_key_user = public_key_user_r.content.decode()
        public_key_registrator_r = requests.get(f'http://0.0.0.0:13451/public')
        public_key_registrator = public_key_registrator_r.content.decode()
        checked_user_sign = check_sign(message_user, RSA.importKey(pub-
lic_key_user), sign_user)
        checked_registrator_sign = check_sign(message_user, RSA.importKey(pub-
lic_key_registrator), sign_registrator)
        if not checked_user_sign:
            raise Exception('Ошибка в подписи пользователя')
        if not checked_registrator_sign:
            raise Exception('Ошибка в подписи регистратора')
```

### check_hash_bulletins.py

```
from db import, select, select_old
from Crypto.Hash import SHA256
import hashlib

def check():
    sql = '''
select * from bulletins
    '''
    result_sql = select(sql)
    result_sql_old = select_old(sql)
```

```
    for row in result_sql_old:
        id_bulletin = row['id']
        old_message = row['message']
        new_message = result_sql[id_bulletin]
        if not new_message:
            raise Exception('Бюллетень не найден')

        hash_old = hashlib.sha256(old_message.encode()).hexdigest()
        hash_new = hashlib.sha256(new_message.encode()).hexdigest()

        if hash_old != hash_new:
            raise Exception('Бюллетени не совпадают')
```

graph.py

```
import matplotlib.pyplot as plt
from datetime import datetime, timedelta
from random import randint
from db import select, select_old

sql = '''
select id from bulletins
    '''
result_sql = select_old(sql)['id']

sql = f'''
select sum* from bulletins
where id not in [{result_sql}]
'''
result_sql = select(sql)['id']

graph = {
}

delta = 0
for row in result_sql:
    qty = graph.get(row['date_time']) or 0
    qty += 1
    graph[row['date_time']] = qty

fig, ax = plt.subplots()


ax.plot(graph.keys(), graph.values(), 'o-b')
ax.set_xlabel('Время',
              fontsize=15
              )
ax.set_ylabel('Количество принятых бюллетеней',
              fontsize=15)

plt.grid(True)
```

# Приложение Г. Клиентское приложение

## client.py

```
from datetime import datetime
from base64 import b64encode
import requests
from tkinter import *
import tkinter.messagebox as tm
from Crypto.PublicKey import RSA
from Crypto.Signature import pkcs1_15
from Crypto.Cipher import PKCS1_OAEP
from Crypto.Hash import SHA256


def encrypt(message):
    encrypt_key = RSA.generate(4096)
    encrypted_message = PKCS1_OAEP.new(encrypt_key).encrypt(message)
    return encrypt_key, encrypted_message


def decrypt(encrypt_key, message):
    return PKCS1_OAEP.new(encrypt_key).decrypt(message)


def sign(encrypted_2_message, private):
    hash_encrypted_2_message = SHA256.new(encrypted_2_message)

    signature = pkcs1_15.new(private).sign(hash_encrypted_2_message)
    return signature

try:
    with open('private.txt', 'r') as f:
        PRIVATE = RSA.importKey(f.read().encode())
        PUBLIC = PRIVATE.publickey()
except FileNotFoundError:
    PRIVATE = RSA.generate(4096)
    PUBLIC = PRIVATE.publickey()

    with open('private.txt', 'w') as f:
        f.write(PRIVATE.export_key().decode())


class LoginFrame(Frame):
    def __init__(self, master):
        super().__init__(master)

        self.empty_label = Label(self)
        self.empty_label.grid(row=0, sticky=E)

        self.label_username = Label(self, text="Логин")
        self.label_password = Label(self, text="Пароль")

        self.entry_username = Entry(self)
        self.entry_password = Entry(self, show="*")

        self.label_username.grid(row=1, sticky=E)
        self.label_password.grid(row=2, sticky=E)
        self.entry_username.grid(row=1, column=1)
        self.entry_password.grid(row=2, column=1)

        # self.empty_label = Label(self)
```

```
        # self.empty_label.grid(row=3, sticky=E)

        self.login_btn = Button(self, text="Войти", command=self.btn_clicked)
        self.login_btn.grid(columnspan=4)

        self.pack()

    def btn_clicked(self):
        username = self.entry_username.get()
        password = self.entry_password.get()
        if not username or not password:
           tm.showerror('Ошибка', 'Неверный логин или пароль!')

        user = self.auth(username, password)
        if user:
            self.destroy()
            StatusVote(self.master, user)

    def auth(self, username, password):
        data = {
            'username': username,
            'password': password,
            'public_key': PUBLIC.export_key().decode(),
        }
        r = requests.post('http://0.0.0.0:13451/auth', json=data)
        try:
            result = r.json()
            if result:
                error = result.get('error_message')
                if error:
                    tm.showinfo(title='Ошибка', message=error, icon='error')
                else:
                    return result
            else:
                tm.showerror('Ошибка', 'Внутренняя ошибка сервера')
        except:
            tm.showerror('Ошибка', 'Внутренняя ошибка сервера')


class StatusVote(Frame):
    def __init__(self, master, user=None):
        super().__init__(master)
        self.user = user
        r = requests.get('http://0.0.0.0:13451/status').json()
        date_time_now = datetime.now()
        status = self.get_status_vote()
        row = 0

        self.empty_label = Label(self)
        self.empty_label.grid(row=row, sticky=E)
        row += 1

        self.label_status = Label(self, text=f'Статус голосования: {status}')
        self.label_status.grid(row=row)
        row += 1

        self.empty_label = Label(self)
        self.empty_label.grid(row=row, sticky=E)
        row += 1

        self.label_time = Label(self, text=f"Текущее время:
{date_time_now.strftime('%d.%m.%y %H:%M:%S')}")
```

```
        self.label_time.grid(row=row, sticky=E)
        row += 1

        self.label_start = Label(self, text=f"Старт голосования:
{r.get('start')}")
        self.label_start.grid(row=row, sticky=E)
        row += 1

        self.label_accepting = Label(self, text=f"Старт подтверждения голосов:
{r.get('accepting')}")
        self.label_accepting.grid(row=row, sticky=E)
        row += 1

        self.label_stop = Label(self, text=f"Остановка подтверждения голосов:
{r.get('stop_voting')}")
        self.label_stop.grid(row=row, sticky=E)
        row += 1

        self.empty_label = Label(self)
        self.empty_label.grid(row=row, sticky=E)
        row += 1

        self.make_buttons(status)

        self.pack()

        self.timer_job = self.master.after(1000*1, self.update_status)

    def make_buttons(self, status):
        if status == 'Голосование начато':
            self.btn = Button(self, text="Проголосовать", com-
mand=self.btn_vote_clicked)
            self.btn.grid(columnspan=7)
            # row += 1
        elif status == 'Процесс подтверждения голосов':
            self.btn = Button(self, text="Подтвердить", command=self.btn_ac-
cept_clicked)
            self.btn.grid(columnspan=7)
            # row += 1

    def update_status(self):
        date_time_now = datetime.now()
        self.label_time['text'] = f"Текущее время:
{date_time_now.strftime('%d.%m.%y %H:%M:%S')}"
        status = self.get_status_vote()
        self.label_status['text'] = f'Статус голосования: {status}'
        self.timer_job = self.master.after(1000 * 1, self.update_status)
        # self.btn.destroy()
        self.make_buttons(status)

    def btn_vote_clicked(self):
        self.master.after_cancel(self.timer_job)
        self.destroy()
        ChoiceCandidate(self.master, self.user)

    def btn_accept_clicked(self):
        tm.showinfo('Успех', 'Голос учтен!')
        self.btn.destroy()

    @staticmethod
    def get_status_vote():
        date_time_now = datetime.now()
```

```
        r = requests.get('http://0.0.0.0:13451/status').json()
        start = datetime.strptime(r.get('start'), '%d.%m.%y %H:%M:%S')
        accepting = datetime.strptime(r.get('accepting'), '%d.%m.%y %H:%M:%S')
        stop_voting = datetime.strptime(r.get('stop_voting'), '%d.%m.%y %H:%M:%S')

        status = 'Голосование еще не начато'
        if accepting > date_time_now > start:
            status = 'Голосование начато'
        elif stop_voting > date_time_now > accepting:
            status = 'Процесс подтверждения голосов'
        elif date_time_now > stop_voting:
            status = 'Голосование завершено'
        return status


class ChoiceCandidate:
    def __init__(self, master, user=None):
        self.user = user
        self.master = master
        self.var = StringVar()
        self.frame = LabelFrame(master, text=f'Привет {user["login"]}, сделай свой
выбор!', padx=50)
        self.frame.pack()
        for candidate in CANDIDATES:
            Radiobutton(self.frame, text=candidate, variable=self.var, value=can-
didate).pack(anchor=W)

        self.btn = Button(master, text='Голосовать', padx=20, pady=5, com-
mand=self.btn_clicked)
        self.btn.pack(pady=10)

    def btn_clicked(self):
        message = str(self.var.get()).encode('utf-8')
        print(message)
        if message:
            encrypt_key, encrypted_message = encrypt(message)
            sign_message = sign(encrypted_message, PRIVATE)

            data = {
                'id': self.user.get('id'),
                # Кодируем в base64, чтобы можно было легко передать по сети
                'sign': b64encode(sign_message).decode(),
                'encrypted_message': b64encode(encrypted_message).decode(),
            }
            result_registrator = requests.post('http://0.0.0.0:13451/vote',
json=data)
            error = result_registrator.json().get('error_message')
            if error:
                tm.showerror('Ошибка', error)
            else:
                data['sign_registrator'] = result_registrator.json().get('sign')

                result_validator = requests.post('http://0.0.0.0:13452/vote',
json=data)
                data['sign_validator'] = result_validator.json().get('sign')

                requests.post('http://0.0.0.0:13452/accept', json={
                    'id': self.user.get('id'),
                    'private': PRIVATE.export_key().decode()
                })

                self.frame.destroy()
```

```
                tm.showinfo('Успех', 'Голос учтен!')
                StatusVote(self.master)
        else:
            tm.showerror(title='Ошибка', message='Пожалуйста, выберите один из
предложенных вариантов', icon='error')


root = Tk()
root.title('Дистанционное электронное голосование')
root.geometry('450x230')
lf = LoginFrame(root)
root.mainloop()
```