Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники.

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

Для информационной системы «Табло авиарейсов»

Преподаватель \_\_\_\_\_\_\_\_\_ Васильев В.С.

подпись

Студент КИ19-06б, 031940348 \_\_\_\_\_\_\_\_\_ Маслов Е.А.

подпись

Красноярск 2020

**СОДЕРЖАНИЕ**

Введение...........................................................................................................3

1. Основание для разработки……………………………………………….3
2. Требования к программе….........................................................................3

2.1. Требования к функциональным характеристикам программы…….3

2.2. Требования к надежности программы……………………………….5

2.3. Требования к информационной и программной совместимости…5

2.4. Условия эксплуатации……………………………………………….5

1. Требования к программной документации……………………………….5
2. Описание входных и выходных данных………………………………..6
3. Диаграмма прецедентов………………………………………………….8
4. Модель предметной области…………………………………………….13
5. Макет интерфейса………………………………………………………..14
6. Стадии и этапы разработки………………………………………………20

8.1. Стадии разработки………………………………………………….20

8.2. Этапы разработки……………………………………………………20

1. Порядок контроля и приёмки…………………………………………..21

ПРИЛОЖЕНИЕ………………………………………………………………22

**Введение**

Информационная система «Табло авиарейсов».

Предназначена для автоматизации работы с пассажирами, позволяет пассажирам просматривать, а также фильтровать рейсы на информационном табло, позволяет диспетчерам вносить изменения на табло авиарейсов (добавлять новые рейсы, удалять и изменять), предупреждать пассажиров о каком либо событии связанным с рейсами (рейс отложен и т.п.).

1. **Основание для разработки**

Разработка информационной системы ведётся в рамках дисциплины “Основы проектирования программного обеспечения”

Согласно учебному плану для получения оценки на экзамене.

1. **Требования к программе**
   1. **Требования к функциональным характеристикам программы**

Анализируя задание, разрабатываемая система поддержки информационного табло, должна обеспечивать следующие функции:

1. Отображение на табло информации о рейсах:
   1. Номер рейса.
   2. Пункт вылета.
   3. Время вылета.
   4. Пункт прилёта.
   5. Время прилёта.

1.6. Статус (отложен, вылетел, прилетел).

1. Изменение в состояние табло (информация о задержке рейса и т. д).
2. Ввод и хранение данных о пассажирах:

3.1. Логин.

3.2. Пароль.

1. Ввод и хранение информации о диспетчерах:

4.1 Логин.

4.2 Пароль.

С данной информационной системой должны работать следующие группы пользователей:

1. Пассажир.
2. Диспетчер.
3. Администратор.

При работе с системой пассажир должен иметь возможность:

1. Просматривать информацию о рейсах.
2. Фильтровать информацию на табло.

При работе с системой диспетчер должен иметь возможность:

1. Вносить изменения в состояние табло, если произошло некоторое событие (например, вылет рейса отложен на N минут, произошла посадка самолета указанного рейса и т. д).

При работе с системой администратор должен иметь возможность:

1. Вносить изменения о пользователях.
2. Вносить изменения в состояние табло, если произошло некоторое событие (например, вылет рейса отложен на N минут, произошла посадка самолета указанного рейса и т. д).
   1. **Требования к надежности программы**

Информационная система должна обеспечивать проверку корректности входных и выходных данных, должна работать устойчиво.

* 1. **Требования к информационной и программной совместимости**

Проект разрабатывается на языке программирования C++ средствами графической библиотеки QT.

* 1. **Условия эксплуатации**

Для эксплуатации информационной системы “Табло авиарейсов” от пользователя требуется умение работать на персональном компьютере.

Программа не требует установленного Framework QT на компьютере пользователя.

1. **Требования к программной документации**

Программная документация для данного проекта делится на техническое задание и пояснительную записку. Программная документация должна полностью отображать возможности и цели программы. Техническое задание объясняет какой будет программа и ставит определенную цель выполнения проекта. Пояснительная записка, в свою очередь, должна полностью объяснять работу программы, содержать в себе макеты интерфейса и описание всех уже прописанных классов, заголовочных файлов, файлов основного программного кода, всю связь между слотами и программным кодом.

1. **Описание входных и выходных данных**

**Входные данные**:

Файл: **users.bin**.

Формат файла: двоичный файл.

Описание: В файле **users.bin** хранятся данные о пользователях.

(Логин, пароль, статус.)

Типы данных:

Login – тип QString, может содержать числа и буквы.

password - тип QString, может содержать числа и буквы.

При попытке войти в систему, эти данные загружаются и сверяются с введёнными пользователем.

Файл: **planes.bin**.

Формат файла: двоичный файл.

Описание: В файле **planes.bin** хранятся данные о рейсах.

(Номер рейса, Пункт вылета, Время вылета, Пункт прилёта, Время прилёта, Статус (отложен, вылетел, прилетел))

Типы данных:

number – QString.

Departure – QString.

Departure\_time – QString.

Arrival – QString.

Arrival\_time – QString.

Status - QString.

В переменных Departure\_time и Arrival\_time хранятся данные о дате и времени вылета, а так же о дате и вылете прилёта. Эти данные хранятся в формате “ДД.ММ.ГГГГ – ЧЧ.ММ”.

Эти данные диспетчер вводит сам.

При открытии главного окна пользователя, эти данные загружаются и выводятся на экран в виде списка рейсов, в таком же формате, как и хранятся в файле.

**Выходные данные**:

Файл: **users.bin**.

Формат файла: двоичный файл.

Описание: В файле **users.bin** хранятся данные о пользователях.

(Логин, пароль.)

Типы данных:

Login – тип QString, может содержать числа и буквы.

password - тип QString, может содержать числа и буквы.

В файл вносятся изменения о пользователях.

Изменять, добавлять, удалять данные в этом файле имеет возможность только администратор.

Файл: **planes.bin**.

Формат файла: двоичный файл.

Описание: В файле **planes.bin** хранятся данные о рейсах.

(Номер рейса, Пункт вылета, Время вылета, Пункт прилёта, Время прилёта, Статус (отложен, вылетел, прилетел))

Типы данных:

number – QString.

Departure – QString.

Departure\_time – QString.

Arrival – QString.

Arrival\_time – QString.

Status - QString.

В переменных Departure\_time и Arrival\_time хранятся данные о дате и времени вылета, а так же о дате и вылете прилёта. Эти данные хранятся в формате “ДД.ММ.ГГГГ – ЧЧ.ММ”.

Эти данные диспетчер вводит сам.

При изменении рейса, удалении рейса и добавления нового рейса, данные в файле обновляются.

Вносить изменения в файл могут диспетчер и администратор.

1. **Диаграмма прецедентов**

Диаграмма прецедентов диспетчера и администратора:

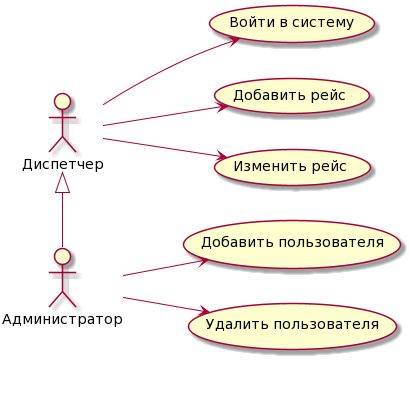


Рисунок 1 – Диаграмма прецедентов диспетчера и администратора

**Название прецедента:**Войти в систему

**Действующее лицо:**Диспетчер

**Цель:**Войти в систему

**Предусловия:**У диспетчера должны быть данные для входа (логин, пароль)

**Главная последовательность:**

1. Диспетчер запускает программу.
2. Диспетчер вводит данные для входа и нажимает кнопку “Войти”. Если данные введённые пользователем корректны, открывается главное окно приложения.

Иначе появляется ошибка о некорректном вводе.

**Название прецедента:**Добавить рейс

**Действующее лицо:**Диспетчер

**Цель:**Добавить новый рейс

**Предусловия:**Диспетчер осуществил вход в систему

**Главная последовательность:**

1. Диспетчер в главном окне переходит в меню диспетчера.
2. Открывается меню диспетчера, диспетчер нажимает кнопку “ Управление рейсами”, после чего переходит в окно управления рейсами.
3. Диспетчер нажимает кнопку “Добавить”.
4. Открывается окно добавления нового рейса.
5. Диспетчер вносит необходимые данные и нажимает

“ Сохранить”.

1. После этого открывается окно управления рейсами.

**Название прецедента:**Внести изменения о рейсах

**Действующее лицо:**Диспетчер

**Цель:** Изменение информации о рейсе

**Предусловия:**Диспетчер осуществил вход в систему

**Главная последовательность:**

1. Диспетчер в главном окне переходит в меню диспетчера.
2. Открывается меню диспетчера, диспетчер нажимает кнопку “ Управление рейсами”, после чего переходит в окно управления рейсами.
3. Диспетчер выбирает конкретный рейс и нажимает кнопку “Редактировать”.
4. Открывается окно редактирования рейса.
5. Диспетчер вносит необходимые поправки в рейс, данные о задержке рейса, вылете рейса и т.д. диспетчер записывает в поле “Статус” и нажимает

“ Сохранить”.

1. После этого открывается окно изменения рейсов.

**Название прецедента:**Добавить пользователя

**Действующее лицо:**Администратор

**Цель:** Добавить нового пользователя

**Предусловия:**Администратор осуществил вход в систему

**Главная последовательность:**

1. Администратор в главном окне нажимает кнопку “Администрирование”.
2. Открывается окно администратора.
3. Администратор нажимает кнопку “Управление пользователями”.
4. Открывается окно со списком всех пользователей.
5. Администратор нажимает кнопку “Добавить”.
6. Открывается окно добавления нового пользователя, в котором администратор вносит необходимые данные, а так же, выбирает статус пользователя(Диспетчер, пассажир). После этого нажимает кнопку “Сохранить”.
7. Открывается окно управления пользователями.

**Название прецедента:**Удалить пользователя

**Действующее лицо:**Администратор

**Цель:** Удалить пользователя

**Предусловия:**Администратор осуществил вход в систему

**Главная последовательность:**

1. Администратор в главном окне нажимает кнопку “Администрирование”.
2. Открывается окно администратора.
3. Администратор нажимает кнопку “Управление пользователями”.
4. Открывается окно со списком всех пользователей.
5. Администратор выделяет пользователя одним нажатием по нему и нажимает кнопку “Удалить”. После чего пользователь удаляется.
6. После открывается окно со списком всех пользователей.

Диаграмма прецедентов пассажира:

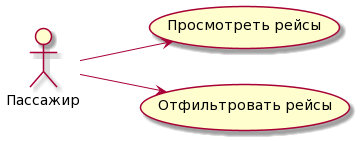


Рисунок 2 – Диаграмма прецедентов пассажира

**Название прецедента:**Просмотреть рейсы

**Действующее лицо:**Пассажир

**Цель:** Посмотреть на список рейсов с целью узнать интересующую информацию

**Предусловия:**-

**Главная последовательность:**

1. Пассажир в главном окне просматривает список рейсов.

**Название прецедента:**Отфильтровать рейсы

**Действующее лицо:**Пассажир

**Цель:** Отфильтровать список рейсов по интересующему параметру.

**Предусловия:**-

**Главная последовательность:**

1. Пассажир в главном окне просматривает список рейсов.
2. В текстовое поле пассажир вводит какой либо параметр, по которому он хочет отфильтровать рейсы.
3. После чего на главном окне выводиться отфильтрованный список рейсов.
4. **. Модель предметной области**

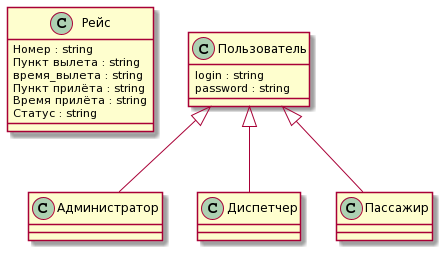
****

Рисунок 3 – Модель предметной области

1. **Макет интерфейса**

Пользовательский интерфейс программы должен содержать:

1. Окно авторизации при запуске программы, а так же окно регистрации, в случае если нет аккаунта в системе, представленные на рисунках 4 и 5 соответственно.

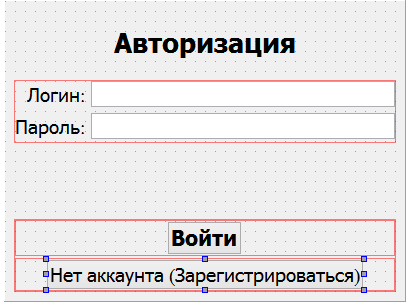
****

Рисунок 4 – Окно авторизации

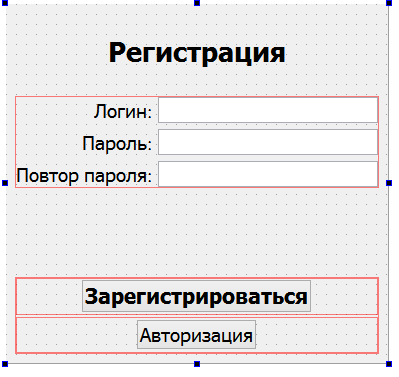
****

Рисунок 5 – Окно регистрации

2. Главное окно со списком рейсов для всех пользователей показано на рисунке 6, у администратора есть кнопка PushButton (Администрирование) в правом верхнем углу, а у других пользователей данная кнопка скрыта.

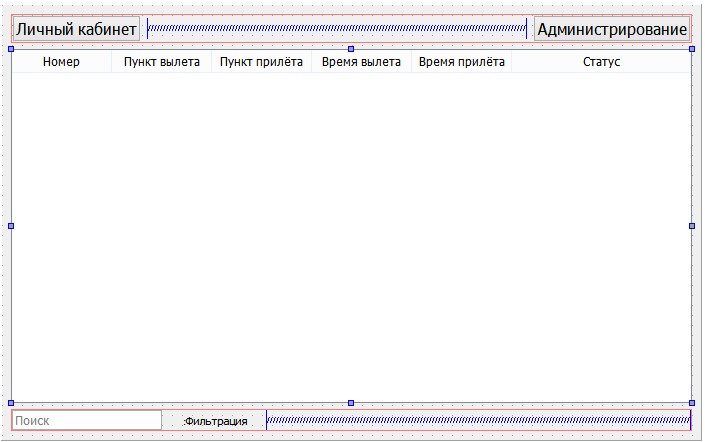
****

Рисунок 6 – Главное окно со списком рейсов

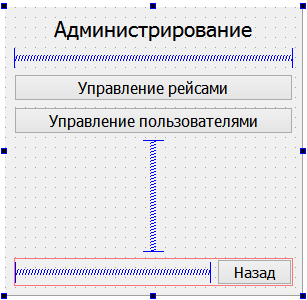
****3. Меню администратора открывается после нажатия на кнопку – Администрирование, окно представлено на рисунке 7.

Рисунок 7 – Меню администратора

4. Меню диспетчера открывается после двойного клика по любому рейсу, окно показано на рисунке 8.

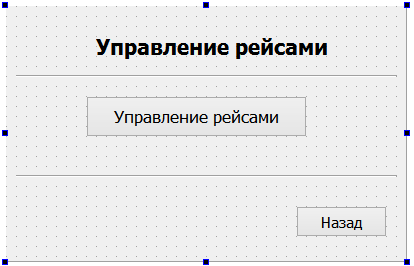
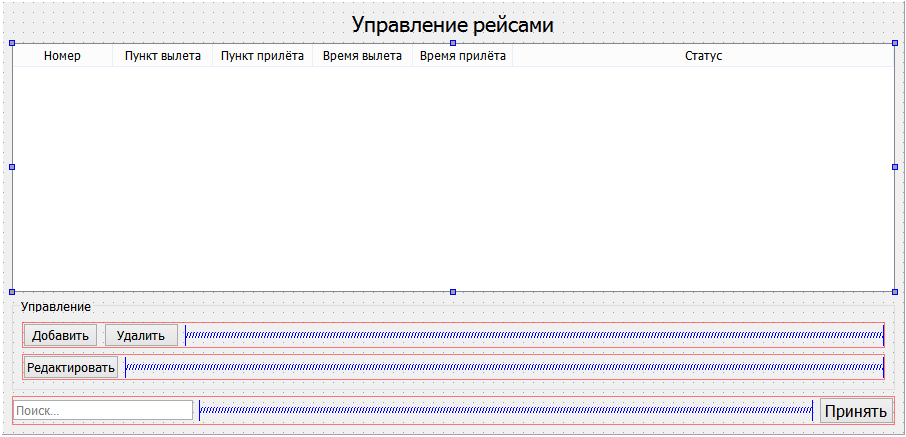
****

Рисунок 8 – Меню диспетчера

5. Как из меню администратора, так и из меню диспетчера можно перейти в окно управления рейсами, окно управления показано на рисунке 9.

В окне управления рейсами можно изменять рейсы (Нужно выделить нужный рейс и нажать кнопку «Редактировать»), так же можно удалять рейсы (Нужно выделить нужный рейс и нажать на кнопку «Удалить»), можно добавлять рейсы (Нужно нажать на кнопку «Добавить»). Окно редактирования/добавления рейса показано на рисунке 10.

****

****

Рисунок 9 – Окно управления рейсами

Рисунок 10 – Окно редактирования/добавления рейса

6. В окно управления пользователями может попасть только администратор, это окно показано на рисунке 11.

В этом окне отображается список всех пользователей системы, администратор может удалить (Нужно выделить рейс и нажать на кнопку «Удалить») или добавить пользователя (Нужно нажать на кнопку «Добавить», после чего откроется окно добавления пользователя как показано на рисунке 12).

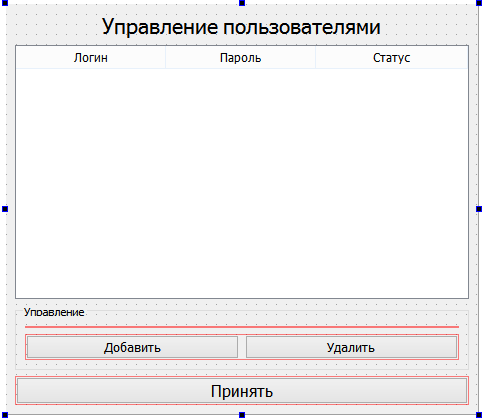
****

Рисунок 11 – Окно управления пользователями

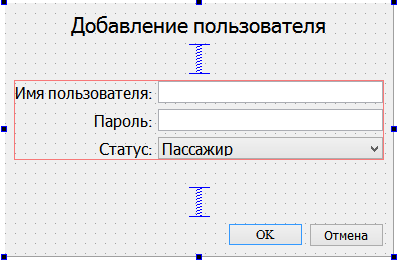
****

Рисунок 12 – Окно добавления пользователя

7. Окно личного кабинета содержит информацию о текущем пользователе который работает с системой, доступ к этому окну имеют все пользователи не зависимо от статуса, окно показано на рисунке 13.

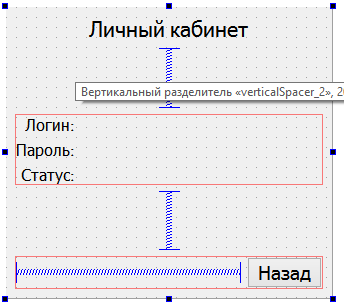
****

Рисунок 13 – Окно личного кабинета

**8. Стадии и этапы разработки**

**8.1. Стадии разработки**

Разработка должна быть проведена в три стадии:

1. Разработка технического задания;
2. Создание приложения.
3. Создание пояснительной записке к проекту;

**8.2. Этапы разработки**

Этапы разработки показаны в таблице 1.

**Таблица 1 - Этапы разработки**

|  |  |
| --- | --- |
| **Этап разработки** | **Сроки** |
| Реализация окна, доступного пассажиру. | 15.10 – 22.10 |
| Реализация окон, доступных администратору. | 22.10 – 19.11 |
| Реализация окон, доступных диспетчеру. | 19.11 – 28.11 |
| Написание пояснительной записки. | * 1. – 3.12 |

**9. Порядок контроля и приёмки**

Производится проверка корректного выполнения функций программы, осуществляется функциональное тестирование программы и визуальная проверка интерфейса.

Проект будет утверждён при корректной работе программы при различных входных и выходных данных, при работе под разными пользователями, а так же при соответствии программы требования технического задания.

**ПРИЛОЖЕНИЕ**