ДОДАТОК А

Графічний матеріал кваліфікаційної роботи

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ
КОМП'ЮТЕРНА ІНЖЕНЕРІЯ ТА УПРАВЛІННЯ, КАФЕДРА ЕОМ

ПІДГОТУВАВ: СТУДЕНТ ГРУПИ СПМ-22-3
ЯКОВЛЄВ А.В.

НАУКОВИЙ КЕРІВНИК: ДОЦ. ІЛЬЇНА І.В.

ІНТЕЛЕКТУАЛЬНА СИСТЕМА ПРИЙНЯТТЯ РІШЕННЯ ДЛЯ ОРГАНІЗАЦІЇ ГУМАНІТАРНОЇ ЛОГІСТИКИ

ХАРКІВ 2024

МЕТА КВАЛІФІКАЦІЙНОЇ РОБОТИ

МЕТОЮ КВАЛІФІКАЦІЙНОЇ РОБОТИ Є **РОЗРОБКА МОДЕЛІ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ДЛЯ ОРГАНІЗАЦІЇ ГУМАНІТАРНОЇ ЛОГІСТИКИ**

ПИТАННЯ, ЯКІ ПОТРІБНО ОПРАЦЮВАТИ

- АНАЛІЗ ІНТЕЛЕКТУАЛЬНИХ ІНФОРМАЦІЙНИХ СИСТЕМ ТА ГАЛУЗЕЙ ЇХ ЗАСТОСУВАННЯ
- РОЗГЛЯНУТИ ІНСТРУМЕНТАРІЙ ДЛЯ СТВОРЕННЯ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ,
- РОЗГЛЯНУТИ ВЖЕ ІСНУЮЧІ ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ТА ЇХ ФУНКЦІОНАЛ
- ПРОЄКТУВАННЯ ТА МОДЕЛЮВАННЯ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ
- ПРОГРАМНА РЕАЛІЗАЦІЯ

АКТУАЛЬНІСТЬ ПИТАННЯ

В СУЧАСНОМУ СВІТІ, ДЕ ГУМАНІТАРНІ КРИЗИ ТА НАДЗВИЧАЙНІ СИТУАЦІЇ СТАЮТЬ НЕВІД'ЄМНОЮ ЧАСТИНОЮ НАШОГО ЖИТТЯ, РОЛЬ ВОЛОНТЕРСЬКИХ ОРГАНІЗАЦІЙ У СФЕРІ ЛОГІСТИКИ СТАЄ ДЕДАЛІ ВАЖЛИВІШОЮ. ЗАБЕЗПЕЧЕННЯ ШВИДКОЇ ТА ЕФЕКТИВНОЇ ДОПОМОГИ У ПОТРІБНИЙ МОМЕНТ ВИМАГАЄ ВИСОКОГО РІВНЯ ОРГАНІЗАЦІЇ ТА ТОЧНОСТІ ПРИ ПРИЙН.

У ЦЬОМУ КОНТЕКСТІ ВИКОРИСТАННЯ СИСТЕМ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ НАБУВАЄ КЛЮЧОВОГО ЗНАЧЕННЯ ДЛЯ КООРДИНАЦІЇ ДІЙ ВОЛОНТЕРІВ У ЛОГІСТИЧНИХ ПРОЦЕСАХЯТТІ РІШЕНЬ

АНАЛІЗ ІНТЕЛЕКТУАЛЬНИХ ІНФОРМАЦІЙНИХ СИСТЕМ ТА ГАЛУЗЕЙ ЇХ ЗАСТОСУВАННЯ

ІІС – ЦЕ КОМП'ЮТЕРНА СИСТЕМА, ЩО СКЛАДАЄТЬСЯ З 5 ОСНОВНИХ ВЗАЄМОДІЮЧИХ КОМПОНЕНТІВ:

- МОВНОЇ ПІДСИСТЕМИ
- ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ
- ПІДСИСТЕМИ УПРАВЛІННЯ ЗНАННЯМИ
- ПІДСИСТЕМИ КЕРУВАННЯ МОДЕЛЯМИ
- ПІДСИСТЕМИ ОБРОБКИ ТА ВИРІШЕННЯ ЗАВДАНЬ

ІНСТРУМЕНТАРІЙ ДЛЯ РОЗРОБКИ СППР

1. АНАЛІЗ І ДЕКОМПОЗИЦІЯ ПРОЦЕСУ ПРИЙНЯТТЯ РІШЕННЯ

2. ФУНКЦІОНАЛЬНЕ ПРОЕКТУВАННЯ ТА РОЗРОБКА СПЕЦИФІКАЦІЙ

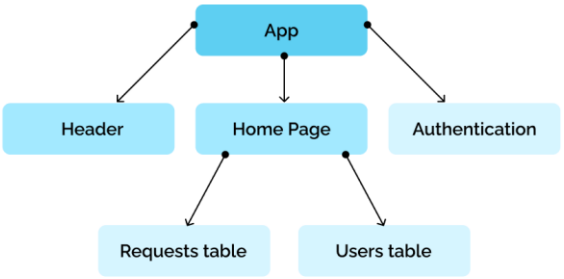
3. РЕАЛІЗАЦІЯ, ВЕРИФІКАЦІЯ ТА СУПРОВІД СИСТЕМИ

ВЖЕ ІСНУЮЧІ ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ТА ЇХ ФУНКЦІОНАЛ

- SUMA
- SAHANA
- HLS TA HELIOS
- DMIS
- LOGISTIX

6

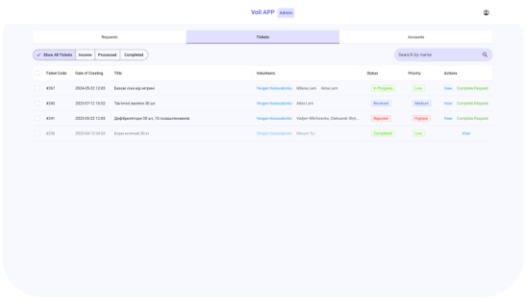
ПРОЄКТУВАННЯ ТА МОДЕЛЮВАННЯ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ

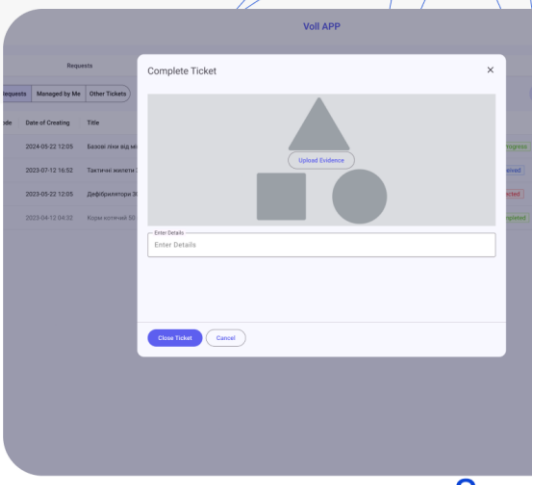


$$\frac{0.4 \times I + 0.3 \times \mathrm{T} + 0.2 \times A + 0.1 \times E}{S + D}$$

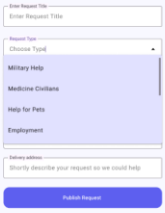
7

ПРОГРАМНА РЕАЛІЗАЦІЯ





ІДЕЇ ДЛЯ ПОКРАЩЕННЯ



- ШТУЧНИЙ ІНТЕЛЕКТ
- ХМАРНІ ОБЧИСЛЕННЯ
- МАШИННЕ НАВЧАННЯ
- ІНТЕРНЕТ РЕЧЕЙ

ВИСНОВКИ

ПІД ЧАС НАПИСАННЯ ДИПЛОМА БУЛИ ОПРАЦЬОВАНІ ПИТАННЯ, ПОВ’ЯЗАНІ З ТЕМОЮ КВАЛІФІКАЦІЙНОЇ РОБОТИ.

1.ПРОАНАЛІЗОВАНО ГАЛУЗІ ЗАСТОСУВАННЯ ТА ВИДИ ІНТЕЛЕКТУАЛЬНИХ СИСТЕМ ПРИЙНЯТТЯ РІШЕНЬ;
2.РОЗРОБЛЕНО ІНТЕЛЕКТУАЛЬНУ СИСТЕМУ ПРИЙНЯТТЯ РІШЕНЬ ДЛЯ ОРГАНІЗАЦІЇ ГУМАНІТАРНОЇ ЛОГІСТИКИ;
3.РОЗРОБЛЕНО ДОДАТОК З ІМПЛЕМЕНТАЦІЄЮ ОСНОВНОЇ ФОРМУЛИ МОДЕЛІ.

ДОДАТОК Б

Лістинг програмного коду

Лістинг Б.1 – Код навчальної платформи, файл package.json

```
  "name": "diploma_master",
  "version": "0.0.0",
  "private": true,
  "type": "module",
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "preview": "vite preview"
  },
  "dependencies": {
    "@quasar/extras": "^1.16.11",
    "pinia": "^2.1.7",
    "quasar": "^2.16.4",
    "vue": "^3.4.21",
    "vue-router": "^4.2.2"
  },
  "devDependencies": {
    "@quasar/vite-plugin": "^1.7.0",
    "@vitejs/plugin-vue": "^5.0.4",
    "sass": "^1.77.4",
    "vite": "^5.2.8"
  }
}
```

Лістинг Б.2 – Код навчальної платформи, файл App.vue

```
<script setup>
import AppHeader from "@/components/AppHeader.vue";
</script>
<template>
  <q-layout view="hHh lpR fFf">
    <AppHeader/>
    <q-page-container class="full-height">
      <router-view/>
    </q-page-container>
  </q-layout>
</template>


<style scoped>
```

```
</style>
```

Лістинг Б.3 – Код навчальної платформи, файл router.ts

```
import {createRouter, createWebHistory} from 'vue-router'


const router = createRouter({
    history: createWebHistory(""),
    routes: [
        {
            path: '/',
            component: () => import("@/views/Home.vue")
        },
        {
            path: '/auth',
            component: () => import('@/views/Auth.vue'),
        },
        {
            path: '/:id',
            component: () => import('@/views/Order.vue'),
        },
        {
            path: '/new-order',
            component: () => import('@/views/AddOrder.vue'),
        },
    ],
})

export default router
```

Лістинг Б.4 – Код навчальної платформи, файл requests.ts

```
import {defineStore} from "pinia";
import {Ref, ref} from "vue";
import {IUser} from "./user";

export enum STATUS {
    IN_PROGRESS = "In progress",
    DONE = 'Done',
    REJECTED = "Rejected",
    PENDING = "Pending",
    WAITING_FOR_ASSIGNMENT = "Waiting for volunteer assignment"
}

export interface IRequest {
    code: string
    userId: string
    title: string,
    description: string,
```

```
    date: string
    status: STATUS
    priority: number
    responsibleVolunteer?: IUser
    assignedVolunteers: IUser[];
}

const date = new Date()
const mockRequests: IRequest[] = [
    {
        code: "1",
        title: "Drugs",
        date: date.toISOString().split('T')[0],
        status: STATUS.IN_PROGRESS,
        priority: 3,
        userId: '3',
        description: "string"
    },
    {
        code: "2",
        title: "Weapon",
        date: date.toISOString().split('T')[0],
        status: STATUS.WAITING_FOR_ASSIGNMENT,
        priority: 7,
        assignedVolunteers: [],
        userId: '3',
        description: "string"

    },
    {
        code: "3",
        title: "Food",
        date: date.toISOString().split('T')[0],
        status: STATUS.PENDING,
        priority: 0,
        userId: '0',
        description: "string"

    },
    {
        code: "4",
        title: "Pickup",
        date: date.toISOString().split('T')[0],
        status: STATUS.DONE, priority: 4, userId: '0',
        description: "string"

    }
]

export const useRequestsStore = defineStore("request", () => {
    const requests: Ref<IRequest[] | null> = ref(mockRequests)

    function getRequest(code) {
```

```
        return requests.value.find(req => req.code === code)

    }

    function assignResponsibleVolunteer(code, user) {
        const request = requests.value.find(req => req.code ===
code)

        const isAlreadyInAssigned =
request.assignedVolunteers.find(vol => vol.id === user.id)
        if (isAlreadyInAssigned) {
            request.assignedVolunteers =
request.assignedVolunteers.filter(vol => vol.id !== user.id)
        }

        request.responsibleVolunteer = user
    }

    function assignVolunteer(code, user) {
        const request = requests.value.find(req => req.code ===
code)

        const isResponsibleVolunteer =
request.responsibleVolunteer?.id === user.id
        if (isResponsibleVolunteer) {
            request.responsibleVolunteer = undefined
        }

        request.assignedVolunteers.push(user)
    }

    function unassign(code, userId) {
        const request = requests.value.find(req => req.code ===
code)

        if (request.responsibleVolunteer.id === userId) {
            request.responsibleVolunteer = undefined
            return
        }
        request.assignedVolunteers =
request.assignedVolunteers.filter(user => user.id !== userId)

    }

    function setPriority(code, priority) {
        const request = requests.value.find(req => req.code ===
code)

        request.priority = priority
        request.status = STATUS.WAITING_FOR_ASSIGNMENT
        request.assignedVolunteers = []
    }
```

```
    function setStatus(code, status) {
        const request = requests.value.find(req => req.code ===
code)
        request.status = status
    }

    function createOrder(title, description, userId) {
        const newCode = requests.value.map(req =>
req.code).reduce((prev, curr, acc = 0) => +curr > +acc ? +curr :
+acc)
        const date = new Date().toISOString().split('T')[0]

        requests.value.push({
            code: (+newCode + 1) + "",
            status: STATUS.PENDING,
            priority: 0,
            date,
            title,
            description,
            userId
        })
    }

    function getUserReq(userId) {
        console.log(userId)
        console.log(requests.value.filter(req => req.userId ===
userId))
        return requests.value.filter(req => req.userId ===
userId)
    }

    function getVolReq(userId) {
        console.log(requests.value.filter(req =>
req.responsibleVolunteer?.id === userId ||
req.assignedVolunteers?.find((vol) => vol?.id === userId)))
        return requests.value.filter(req =>
req.responsibleVolunteer?.id === userId ||
req.assignedVolunteers?.find((vol) => vol?.id === userId))
    }

    return {
        requests,
        assignVolunteer,
        assignResponsibleVolunteer,
        getVolReq,
        unassign,
        setStatus,
        createOrder,
        setPriority,
        getRequest,
        getUserReq
    }
})
```

Лістинг Б.5 – Код навчальної платформи, файл user.ts

```
import {defineStore} from "pinia";
import {computed, Ref, ref} from "vue";
import {useRouter} from "vue-router";

export enum ROLES {
    SUPER_ADMIN = "Super admin",
    ADMIN = "Admin",
    VOLUNTEER = "Volunteer",
    USER = "User"
}

export interface IUser {
    id: string
    name: string
    email: string
    phone: string
    city: string
    password: string
    role: ROLES
}

const mockUsers: IUser[] = [
    {
        id: '0',
        name: "SupAdmin",
        // email: "superAdmin@test.com",
        email: "1",
        password: "1",
        role: ROLES.SUPER_ADMIN,
        phone: "+333333333333",
        city: "Kharkiv"

    },
    {
        id: '1',
        name: "Admin",
        email: "admin@test.com",
        password: "1",
        role: ROLES.ADMIN,
        phone: "+333333333333",
        city: "Kyiv"
    },

    {
        id: '15',
        name: "Volunteer15",
        email: "volunteer15@test.com",
        password: "1",
        role: ROLES.VOLUNTEER,
        phone: "+232222222222",
```

```
        city: "Kyiv"
    },
    {
        id: '2',
        name: "Volunteer",
        email: "volunteer@test.com",
        password: "1",
        role: ROLES.VOLUNTEER,
        phone: "+111111111111111",
        city: "Kharkiv"
    },
    {
        id: '3',
        name: "John Doe",
        email: "jainDoe@test.com",
        password: "1",
        role: ROLES.USER,
        phone: "+4444444444444",
        city: "Kyiv"
    }
]


export const useUserStore = defineStore("user", () => {
    const router = useRouter()

    const users = ref(mockUsers)
    const user: Ref<IUser | null> = ref(mockUsers[0])


    function login(email: string, password: string) {
        user.value = mockUsers.find((user) =>
            (user.email === email && user.password === password)
|| null
        )
        if (user.value) {
            router.replace("/")
        }
    }

    function logout() {
        user.value = null
        console.log(user.value)
    }

    function changeRole(id, role) {
        users.value.find(user => user.id === id).role = role
    }

    const isAdmin = computed(() => user.value.role ===
ROLES.ADMIN || user.value.role === ROLES.SUPER_ADMIN)

    function getUser(id) {
```

```
        users.value.find(user => {
            return user.id === id
        })
        return users.value.find(user => user.id === id)
    }

    return {user, isAdmin, users, login, logout, changeRole,
getUser}
})
```

Лістинг Б.6 – Код навчальної платформи, файл main.ts

```
// FILE: main.js

import {createApp} from 'vue'
import {Quasar} from 'quasar'

// Import icon libraries
import '@quasar/extras/material-icons/material-icons.css'

// Import Quasar css
import 'quasar/src/css/index.sass'

// Assumes your root component is App.vue
// and placed in same folder as main.js
// @ts-ignore
import App from './App.vue'
// @ts-ignore
import router from "@/router/index.ts";
import {createPinia} from "pinia";

const app = createApp(App)
const pinia = createPinia()

app.use(pinia)
app.use(Quasar, {
    plugins: {}, // import Quasar plugins and add here
})
app.use(router)

// Assumes you have a <div id="app"></div> in your index.html
app.mount('#app')
```

Лістинг Б.7 – Код навчальної платформи, файл Header.vue

```
<script setup lang="ts">
import {ref} from "vue";
import {useUserStore} from "@/store/user";
```

```
import {storeToRefs} from "pinia";
import {useRouter} from "vue-router";

const userStore = useUserStore()
const {user} = storeToRefs(userStore)
const {logout} = userStore

const router = useRouter()
const rightDrawerOpen = ref(false)

</script>

<template>
  <q-header elevated class="bg-primary text-white">
    <q-toolbar>
      <q-toolbar-title class="cursor-pointer"
@click="router.push('/')">
        Vol app
      </q-toolbar-title>
      <q-btn v-if="user" class="q-mr-md" color="white" flat>
        <router-link class="text-white" to="/new-order">Create
request</router-link>
      </q-btn>
      <div v-if="user">
        <q-btn class="q-mr-md" color="white" flat>
          <router-link @click="logout" class="text-white"
to="/">Logout</router-link>

        </q-btn>
      </div>
      <q-btn v-else class="q-mr-md" color="white" flat>
        <router-link class="text-white"
to="/auth">Login</router-link>

      </q-btn>

    </q-toolbar>
  </q-header>


</template>

<style scoped>

</style>
```

Лістинг Б.8 – Код навчальної платформи, файл Home.vue

```
<script lang="ts" setup>

import {ROLES, useUserStore} from "@/store/user";
```

```
import UsersTable from "@/components/UsersTable.vue";
import {useRequestsStore} from "@/store/requests";
import {computed} from "vue";
import {storeToRefs} from "pinia";
import RequestsTable from "@/components/RequestsTable.vue";

const userStore = useUserStore()

const {user, users} = storeToRefs(userStore)

const {getUserReq, getVolReq, requests} = useRequestsStore()
const userReq = computed(() => getUserReq(user.value.id))
const volunteerReq = computed(() =>
removeDuplicates([...getVolReq(user.value.id),
...userReq.value]))

function removeDuplicates(array) {
  const jsonObject = array.map(JSON.stringify);
  const uniqueSet = new Set(jsonObject);
  const uniqueArray = Array.from(uniqueSet).map(JSON.parse);
  return uniqueArray
}
</script>

<template>
  <div v-if="user">
    <div v-if="user.role===ROLES.SUPER_ADMIN ||
user.role===ROLES.ADMIN">
      <RequestsTable :requests="requests"/>
      <UsersTable class="q-ma-md" :request="undefined"
:users="users"/>
    </div>
    <div v-else-if="user.role===ROLES.USER ">
      <RequestsTable v-if="userReq" :requests="userReq"/>
    </div>
    <div v-else-if="user.role===ROLES.VOLUNTEER">
      <RequestsTable v-if="volunteerReq || userReq"
:requests="volunteerReq"/>
    </div>
  </div>
  <div v-else>Login to continue</div>
</template>

<style scoped>

</style>
```

Лістинг Б.9 – Код навчальної платформи, файл Order.vue

```
<script lang="ts" setup>
```

```
import {useRouter} from "vue-router";
import {IRequest, STATUS, useRequestsStore} from
"@/store/requests";
import {computed, onMounted, ref} from "vue";
import {IUser, ROLES, useUserStore} from "@/store/user";
import UsersTable from "@/components/UsersTable.vue";
import RateOrderDialog from "@/components/RateOrderDialog.vue";

const router = useRouter()

const {getRequest, setStatus} = useRequestsStore()
const {user, users, getUser} = useUserStore()

const request = ref<IRequest>()
const requestUser = ref<IUser>()
const isEstimateOpened = ref(false)

const assignedVolunteers = computed(() => {
  const result = [request.value.responsibleVolunteer,
...request.value.assignedVolunteers]
  return result
})

const availableVolunteers = computed(() => {
  const result = users.filter(user => user.role ===
ROLES.VOLUNTEER)
  return result
})

onMounted(() => {
  request.value =
getRequest(router.currentRoute.value.params.id)
  requestUser.value = getUser(request.value.userId)
})

function openEstimate() {
  isEstimateOpened.value = true
}

function closeEstimate() {
  isEstimateOpened.value = false
}
</script>

<template>
  <div v-if="requestUser&&request" class="q-mx-md">
    <q-btn class="q-my-sm" flat @click="()=>router.back()"><
Back</q-btn>
    <h6 class="q-my-sm">{{ `${request.title} / #${request.code}`
}}</h6>
    <div v-if="!!request.priority">
      <b>Priority:</b>
      <div>{{ request.priority }}</div>
```

```
    </div>
    <div v-if="!!request.priority">
      <b>Status:</b>
      <div>{{ request.status }}</div>
    </div>
    <b>Description:</b>
    <div>{{ request.description }}</div>
    <h6 class="q-my-sm">User info</h6>
    <div><b>Name:</b> {{ requestUser.name }}</div>
    <div><b>Role:</b> {{ requestUser.role }}</div>
    <div><b>Email:</b> {{ requestUser.email }}</div>
    <div><b>Phone:</b> {{ requestUser.phone }}</div>
    <div><b>City:</b> {{ requestUser.city }}</div>
    <div
        v-if="(user.role===ROLES.SUPER_ADMIN ||
user.role===ROLES.ADMIN)&&request.status!==STATUS.IN_PROGRESS &&
request.status!==STATUS.DONE&&
request.status!==STATUS.REJECTED">
      <h6 class="q-my-sm">Actions</h6>
      <q-btn color="primary" flat @click="openEstimate">Estimate
request</q-btn>
      <RateOrderDialog :request="request" @close-
estimate="closeEstimate" v-if="isEstimateOpened"/>
      <div
          v-if="request.status===STATUS.WAITING_FOR_ASSIGNMENT">
        <b class="q-my-sm">Add volunteer</b>
        <UsersTable class="q-mt-sm" :users="availableVolunteers"
:request="request">

        </UsersTable>
      </div>
    </div>

    <div v-if="request.responsibleVolunteer">
      <h6 class="q-my-sm">Assigned volunteer</h6>
      <UsersTable :users="assignedVolunteers"
:request="request"></UsersTable>
    </div>
    <div class="row justify-end" v-
if="request.responsibleVolunteer">
      <q-btn v-
if="request.status!==STATUS.DONE&&request.status!==STATUS.IN_PRO
GRESS&&request.status!==STATUS.REJECTED"
             class="q-ma-md" color="primary" flat

@click="setStatus(request.code,STATUS.IN_PROGRESS)">Start
      </q-btn>
    </div>
    <q-btn
        v-if="(user.role===ROLES.ADMIN
||user.role===ROLES.SUPER_ADMIN)&&(request.status!==STATUS.DONE&
&request.status!==STATUS.REJECTED)"
        class="q-ma-md" color="primary" flat
```

```
        @click="setStatus(request.code,STATUS.REJECTED)">Reject
    </q-btn>
    <q-btn
        v-
if="(request?.responsibleVolunteer?.id===user?.id)||(user.role==
=ROLES.ADMIN||user.role===ROLES.SUPER_ADMIN&&request.status===ST
ATUS.IN_PROGRESS)"
        class="q-ma-md"
        color="primary" flat
        @click="setStatus(request.code,STATUS.DONE)">Done
    </q-btn>

  </div>
</template>

<style scoped>

</style>
```

Лістинг Б.10 – Код навчальної платформи, файл AddOrder.vue

```
<script setup>

import {ref} from "vue";
import {useRequestsStore} from "@/store/requests";
import {useUserStore} from "@/store/user";
import {useRouter} from "vue-router";

const router = useRouter()
const {createOrder} = useRequestsStore()
const {user} = useUserStore()

const title = ref("")
const desc = ref("")

function createRequest() {
  if (title.value && desc.value) {
    createOrder(title.value, desc.value, user.id)
    router.push("/")
  }
}
</script>

<template>
  <div class="row justify-center items-center full-height">
    <q-card style="width: 500px">
      <q-card-section>
        <q-input label="Title" v-model="title"></q-input>
        <q-input type="textarea" label="Description" v-
model="desc"></q-input>
```

```
        <q-card-actions class="row justify-end">
          <q-btn color="primary" @click="createRequest"
flat>Create</q-btn>
        </q-card-actions>
      </q-card-section>
    </q-card>
  </div>

</template>

<style scoped>

</style>
```

Лістинг Б.11 – Код навчальної платформи, файл Auth.vue

```
<script setup lang="ts">
import {ref} from "vue";
import {useUserStore} from "@/store/user";

const {isAddUser} = defineProps<{ isAddUser: boolean }>()

const {login} = useUserStore()

const isSingIn = ref(false)
const email = ref("")
const password = ref("")

function toggleIsSignIn() {
  isSingIn.value = !isSingIn.value
}


function signIn() {
  //TBD sing in functionality
}
</script>

<template>
  <div class="column justify-center content-center q-pt-lg">
    <h6 class="q-pa-lg">
      Please {{ isSingIn ? "sign in" : "login" }} to start using
app
    </h6>
    <q-input label="Email" v-model="email"/>
    <q-input label="Password" v-model="password"/>
    <div class="row justify-end content-center q-mt-md">
      <template v-if="isSingIn">
        <q-btn @click="toggleIsSignIn">Login</q-btn>
        <q-btn @click="signIn" class="q-ml-md">Sign in</q-btn>
      </template>
```

```
      <template v-else>
        <q-btn @click="toggleIsSignIn" class="q-mr-md">Sign
in</q-btn>
        <q-btn @click="()=>login(email, password)">Login</q-btn>
      </template>
    </div>
  </div>
</template>

<style scoped>
</style>
```

Лістинг Б.12 – Код навчальної платформи, файл UsersTable.vue

```
<script setup lang="ts">
import {IUser, ROLES, useUserStore} from "@/store/user";
import {useRouter} from "vue-router";
import {ref} from "vue";
import {storeToRefs} from "pinia";
import {IRequest, STATUS, useRequestsStore} from
"@/store/requests";

const columns = [
  {
    name: 'id',
    required: true,
    label: 'id',
    align: 'left',
    field: 'id',
    sortable: true
  },
  {
    name: 'name',
    required: true,
    label: 'Name',
    align: 'left',
    field: 'name',
    sortable: true
  },
  {
    name: 'city', label: 'City', field: 'city', align: 'left',
    sortable: true
  },
  {
    name: 'phone', align: 'left',
    label: 'Phone', field: 'phone', sortable: true
  },
  {
    name: 'email', align: 'left',
    label: 'Email', field: 'email', sortable: true
  },
```

```
  {
    name: 'role', label: 'Role', align: 'left',
    field: 'role'
  },
  {
    name: 'actions', label: 'Actions', align: 'right',
    field: 'actions'
  },
]

const {users, request} = defineProps<{ users: IUser[], request:
IRequest | null }>()

const router = useRouter()

const {changeRole} = useUserStore()
const {assignResponsibleVolunteer, assignVolunteer, unassign} =
useRequestsStore()

const {isAdmin} = storeToRefs(useUserStore())

const filterUsers = ref('')

function isAssing(row: string) {
  const isEstimated = request.status ===
STATUS.WAITING_FOR_ASSIGNMENT
  const isAlreadyInAssigned =
request.assignedVolunteers.find(vol => vol.id === row)
  const isResponsibleVolunteer =
request.responsibleVolunteer?.id === row

  return isEstimated && (!isAlreadyInAssigned &&
!isResponsibleVolunteer)
}

function isUnassing(row: string) {
  const isResponsibleVolunteer =
request.responsibleVolunteer?.id === row
  if (isResponsibleVolunteer) {
    return true
  }
  const isEstimated = request.status ===
STATUS.WAITING_FOR_ASSIGNMENT
  const isAlreadyInAssigned =
request.assignedVolunteers.find(vol => vol.id === row)

  return isEstimated && isAlreadyInAssigned
}

function isAssingResponsible(row: string) {
  const isEstimated = request.status ===
STATUS.WAITING_FOR_ASSIGNMENT
  const isResponsibleVolunteer =
```

```
request.responsibleVolunteer?.id === row

  return isEstimated && !isResponsibleVolunteer
}


function redirectToUserPage(_, row) {
  router.push(`/${row.code}`)
}

function clickAssign(userId) {
  const user = users.find(item => item.id === userId)
  assignVolunteer(request.code, user)
}

function clickAssignResponsible(userId) {
  const user = users.find(item => item.id === userId)
  assignResponsibleVolunteer(request.code, user)
}
</script>

<template>
  <q-table
      title="Users"
      :rows="users"
      :columns="columns"
      :filter="filterUsers"
      row-key="name"
  >
    <template v-slot:top-right>
      <q-input borderless dense debounce="300" v-
model="filterUsers" placeholder="Search">
        <template v-slot:append>
          <q-icon name="search"/>
        </template>
      </q-input>
    </template>
    <template #body-cell-role="scope">
      <q-td :props="scope">
        {{ request?.responsibleVolunteer && scope.pageIndex ===
0 ? 'Responsible volunteer' : scope.row.role }}
      </q-td>
    </template>

    <template #body-cell-actions="scope">
      <q-td :props="scope">
        <div v-if="!request">
          <q-btn v-if="scope.row.role===ROLES.USER&&isAdmin"
                 flat color="primary"
                 size="sm"

@click="changeRole(scope.row.id,ROLES.VOLUNTEER)">
            Upgrade to volunteer
```

```
          </q-btn>
          <q-btn
              v-else-
if="scope.row.role===ROLES.VOLUNTEER&&isAdmin"
              flat color="primary"
              size="sm"
              @click="changeRole(scope.row.id,ROLES.USER)">
            Downgrade to user
          </q-btn>
          <q-btn
              v-if="scope.row.role===ROLES.USER
||scope.row.role===ROLES.VOLUNTEER"
              color="primary"
              size="sm"
              @click="changeRole(scope.row.id,ROLES.ADMIN)">
            Give admin
          </q-btn>
        </div>
        <div v-else>
          <div v-if="request.status!==STATUS.IN_PROGRESS &&
request.status!==STATUS.DONE">
            <q-btn
                v-if="isAssingResponsible(scope.row.id)"
                class="block"
                size="sm"
                color="primary"
                flat

@click="()=>clickAssignResponsible(scope.row.id)">
              Assign as responsible for delivery
            </q-btn>
            <q-btn
                v-if="isAssing(scope.row.id)"
                class="block" size="sm"
                color="primary" flat
                @click="()=>clickAssign(scope.row.id)">
              Assign
            </q-btn>
            <q-btn
                v-if="isUnassing(scope.row.id)"
                flat color="primary"
                size="sm"
                @click="()=>unassign(request.code,scope.row.id)"
            >
              Unassign
            </q-btn>
          </div>
          <div v-else class="text-grey-6">No actions
available</div>

        </div>
      </q-td>
    </template>
```

```
  </q-table>
</template>

<style scoped>

</style>
```

Лістинг А.13 – Код навчальної платформи, файл OrdersTable.vue

```
<script setup lang="ts">
import {IRequest} from "@/store/requests";
import {useRouter} from "vue-router";
import {ref} from "vue";

const columns = [
  {
    name: 'code',
    required: true,
    label: 'Request code',
    align: 'left',
    field: 'code',
    sortable: true
  },
  {
    name: 'priority',
    required: true,
    label: 'Priority',
    align: 'left',
    field: 'priority',
    sortable: true
  },
  {
    name: 'date', label: 'Date of requesting', field: 'date',
align: 'left',
    sortable: true
  },
  {
    name: 'title', align: 'left',
    label: 'Title', field: 'title', sortable: true
  },
  {
    name: 'status', label: 'Status', align: 'left',
    field: 'status'
  },
]
const router = useRouter()

const {requests} = defineProps<{ requests: IRequest[] }>()

function redirectToRequestPage(_, row) {
  router.push(`/${row.code}`)
```

```
}

const filterRequests = ref('')

</script>

<template>
  <div class="q-pa-md">
    <q-table
        title="Requests"
        :rows="requests"
        :columns="columns"
        row-key="name"
        @rowClick="redirectToRequestPage"
        :filter="filterRequests"
    >
      <template v-slot:top-right>
        <q-input borderless dense debounce="300" v-
model="filterRequests" placeholder="Search">
          <template v-slot:append>
            <q-icon name="search"/>
          </template>
        </q-input>
      </template>
    </q-table>
  </div>
</template>

<style scoped>

</style>
```

Лістинг Б.14 – Код навчальної платформи, файл RateOe.vue

```
<script lang="ts" setup>
import {ref} from "vue";
import {IRequest, useRequestsStore} from "@/store/requests";

const {request} = defineProps<{ request: IRequest }>()
const emit = defineEmits(['closeEstimate'])

const {setPriority} = useRequestsStore()

const importance = ref(0)
const terms = ref(0)
const efficiency = ref(0)
const availability = ref(0)
const specifications = ref(0)
const distance = ref(0)

const isApproveClicked = ref(false)
```

```
function approve() {
  isApproveClicked.value = true
  if (!importance.value || !terms.value || !efficiency.value ||
!availability.value || !specifications.value || !distance.value)
{
    return
  }
  const priority = (importance.value * 0.4) + (terms.value *
0.3) + (efficiency.value * 0.2) + (availability.value * 0.1) /
distance.value + specifications.value
  setPriority(request.code, priority)
  emit('closeEstimate')
}
</script>

<template>
  <q-card class="q-my-md">
    <q-card-section>
      <h6 class="q-ma-none">Please estimate order by
criteria:</h6>
    </q-card-section>
    <q-card-section class="q-pt-none row items-center justify-
between">
      <div>Importance:</div>
      <q-rating
          v-model="importance"
          size="2em"
          :max="5"
          color="primary"
      />
    </q-card-section>
    <q-card-section class="q-pt-none row items-center justify-
between">
      <div>Terms:</div>
      <q-rating
          v-model="terms"
          size="2em"
          :max="5"
          color="primary"
      />
    </q-card-section>
    <q-card-section class="q-pt-none row items-center justify-
between">
      <div>Efficiency:</div>
      <q-rating
          v-model="efficiency"
          size="2em"
          :max="5"
          color="primary"
      />
    </q-card-section>
    <q-card-section class="q-pt-none row items-center justify-
between">
```

```
      <div>Availability:</div>
      <q-rating
          v-model="availability"
          size="2em"
          :max="5"
          color="primary"
      />
    </q-card-section>
    <q-card-section class="q-pt-none row items-center justify-
between">
      <div>Specifications:</div>
      <q-rating
          v-model="specifications"
          size="2em"
          :max="5"
          color="primary"
      />
    </q-card-section>
    <q-card-section class="q-pt-none row items-center justify-
between">
      <div>Distance:</div>
      <q-rating
          v-model="distance"
          size="2em"
          :max="5"
          color="primary"
      />
    </q-card-section>

    <div v-if="isApproveClicked">
      <q-card-section
          v-
if="!importance||!terms||!efficiency||!availability||!specificat
ions||!distance"
          class="q-pt-none text-red row justify-start">
        Please fill all fields
      </q-card-section>
    </div>
    <q-card-actions class="column items-end">
      <div class="row justify-end">
        <q-btn @click="()=>emit('closeEstimate')"
flat>Cancel</q-btn>
        <q-btn @click="approve" flat>Approve</q-btn>
      </div>
    </q-card-actions>


  </q-card>
</template>

<style scoped>

</style>
```

Лістинг Б.15 – Код навчальної платформи, файл index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="icon" href="/favicon.ico">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Vite App</title>
</head>
<body>
<div id="app"></div>
<script type="module" src="/src/main.ts"></script>
</body>
</html>
```

Лістинг Б.16 – Код навчальної платформи, файл vite.config.ts

```
import { fileURLToPath, URL } from 'node:url'

import { defineConfig } from 'vite'
import vue from '@vitejs/plugin-vue'


import { quasar, transformAssetUrls } from '@quasar/vite-plugin'

// https://vitejs.dev/config/
export default defineConfig({
  plugins: [
    vue({
      template: { transformAssetUrls }
    }),

    // @quasar/plugin-vite options list:
    // https://github.com/quasarframework/quasar/blob/dev/vite-
plugin/index.d.ts
    quasar({
      sassVariables: 'src/quasar-variables.sass'
    })
  ],
  resolve: {
    alias: {
      '@': fileURLToPath(new URL('./src', import.meta.url))
    }
  }
})
```