ДОДАТОК А

Серверна частина сайту

Лістинг А.1 – Програмний код файлу routes.py

```
import requests
from flask import render_template, redirect, url_for,
flash, request
from flask_login import login_user, logout_user,
login_required, current_user
from werkzeug.security import generate_password_hash,
check_password_hash
from datetime import datetime
from extensions import db
from models import User, Expense, Income, Field,
FieldHistory, Employee, SalaryPayment,
Machinery,Maintenance
from forms import RegisterForm, LoginForm, ExpenseForm,
IncomeForm, FieldForm, EmployeeForm, SalaryPaymentForm
from predict_yield import predict_yield
from weather_utils import fetch_weather_forecast_summary
from ai_features import explain_prediction
def init_routes(app):
    @app.route('/')
    def index():
        if current_user.is_authenticated:

            field_count =
Field.query.filter_by(owner_id=current_user.id).count()
            employee_count = Employee.query.count()
            income_total =
db.session.query(db.func.sum(Income.amount)).filter_by(use
r_id=current_user.id).scalar() or 0
            expense_total =
db.session.query(db.func.sum(Expense.amount)).filter_by(
```

Продовження лістингу А.1

```
                user_id=current_user.id).scalar() or 0

            recent_fields =
Field.query.filter_by(owner_id=current_user.id).order_by(F
ield.id.desc()).limit(3)
            recent_incomes =
Income.query.filter_by(user_id=current_user.id).order_by(I
ncome.id.desc()).limit(3)

            recent_events = [
                                f" : {f.name}" for f in
recent_fields
                            ] + [
                                f": {i.amount} " for i in
recent_incomes
                            ]

            return render_template(
                'index.html',
                field_count=field_count,
                employee_count=employee_count,
                income_total=income_total,
                expense_total=expense_total,
                recent_events=recent_events
            )
        else:

            return render_template('welcome.html')

    @app.route('/register', methods=['GET', 'POST'])
    def register():
        form = RegisterForm()
        if form.validate_on_submit():
            user = User(username=form.username.data)
```

Продовження лістингу А.1

```
            user.password_hash =
generate_password_hash(form.password.data)
            db.session.add(user)
            db.session.commit()
            flash(" .", "success")
            return redirect(url_for('login'))
        return render_template('register.html', form=form)

    @app.route('/login', methods=['GET', 'POST'])
    def login():
        form = LoginForm()
        if form.validate_on_submit():
            user =
User.query.filter_by(username=form.username.data).first()
            if user and
check_password_hash(user.password_hash,
form.password.data):
                login_user(user)
                return redirect(url_for('index'))
            flash("    .", "danger")
        return render_template('login.html', form=form)
    @app.route('/logout')
    @login_required
    def logout():
        logout_user()
        return redirect(url_for('index'))
    @app.route('/add-expense', methods=['GET', 'POST'])
    @login_required
    def add_expense():
        form = ExpenseForm()
        form.field_id.choices = [(f.id, f.name) for f in
Field.query.filter_by(owner_id=current_user.id).all()]
        if form.validate_on_submit():
```

Продовження лістингу А.1

```
            expense = Expense(
                amount=form.amount.data,
                category=form.category.data,
                field_id=form.field_id.data,
                user_id=current_user.id
            )
            db.session.add(expense)

            field = Field.query.get(form.field_id.data)
            year = field.sowing_date.year if field and
field.sowing_date else datetime.now().year
            history =
FieldHistory.query.filter_by(field_id=field.id,
year=year).first()

            if history:
                amount = form.amount.data
                if form.category.data == 'seed':
                    history.seed_cost = (history.seed_cost
or 0) + amount
                elif form.category.data == 'fertilizer':
                    history.fertilizer_cost =
(history.fertilizer_cost or 0) + amount
                elif form.category.data == 'herbicide':
                    history.herbicide_cost =
(history.herbicide_cost or 0) + amount
                elif form.category.data == 'fuel':
                    history.fuel_cost = (history.fuel_cost
or 0) + amount
                else:
                    history.other_cost =
(history.other_cost or 0) + amount
                history.total_cost = sum([
```

Продовження лістингу А.1

```
                    history.seed_cost or 0,
                    history.fertilizer_cost or 0,
                    history.herbicide_cost or 0,
                    history.fuel_cost or 0,
                    history.other_cost or 0
                ])

            db.session.commit()
            flash("    .", "success")
            return redirect(url_for('index'))

        return render_template('add_expense.html',
form=form)

    @app.route('/add-income', methods=['GET', 'POST'])
    @login_required
    def add_income():
        form = IncomeForm()
        if form.validate_on_submit():
            income = Income(
                source=form.source.data,
                amount=form.amount.data,
                user_id=current_user.id
            )
            db.session.add(income)
            db.session.commit()
            flash("  .", "success")
            return redirect(url_for('index'))
        return render_template('add_income.html',
form=form)

    @app.route('/fields')
    @login_required
    def view_fields():
```

Продовження лістингу А.1

```
        fields =
Field.query.filter_by(owner_id=current_user.id).all()
        return render_template('fields.html',
fields=fields)
    @app.route('/add-field', methods=['GET', 'POST'])
    @login_required
    def add_field():
        form = FieldForm()
        if form.validate_on_submit():
            field = Field(
                name=form.name.data,
                area=form.area.data,
                soil_type=form.soil_type.data,
                location=form.location.data,
                current_crop=form.current_crop.data,
                sowing_date=form.sowing_date.data,
                latitude=form.latitude.data,
                longitude=form.longitude.data,
                harvest_status=form.harvest_status.data,
                harvest_date=form.harvest_date.data,
                owner_id=current_user.id
            )
            db.session.add(field)
            db.session.flush()
            history = FieldHistory(
                field_id=field.id,
                crop_name=field.current_crop,
                sowing_date=field.sowing_date,
                harvest_date=field.harvest_date,
                year=field.sowing_date.year if
field.sowing_date else datetime.now().year
            )
            db.session.add(history)
```

Продовження лістингу А.1

```
            db.session.commit()
            flash("    .", "success")
            return redirect(url_for('view_fields'))
        return render_template('add_field.html',
form=form)
    @app.route('/field/<int:field_id>/delete',
methods=['POST'])
    @login_required
    def delete_field(field_id):
        field = Field.query.get_or_404(field_id)
        if field.owner_id != current_user.id:
            flash("    .", "danger")
            return redirect(url_for('view_fields'))
        db.session.delete(field)
        db.session.commit()
        flash("  .", "success")
        return redirect(url_for('view_fields'))

    def reverse_geocode(lat, lon):
        try:
            url =
"https://nominatim.openstreetmap.org/reverse"
            params = {
                "lat": lat,
                "lon": lon,
                "format": "json",
                "zoom": 10,
                "addressdetails": 1
            }
            headers = {"User-Agent": "farm-ai-app"}
            response = requests.get(url, params=params,
headers=headers)
            data = response.json()
```

Продовження лістингу А.1

```
            address = data.get("address", {})
            return address.get("city") or
address.get("town") or address.get("village") or ""
        except Exception as e:
            print(" reverse geocode error:", e)
            return ""

    @app.route("/field/<int:field_id>")
    @login_required
    def field_detail(field_id):
        field = Field.query.get_or_404(field_id)
        if field.owner_id != current_user.id:
            flash("    .", "danger")
            return redirect(url_for('view_fields'))

        from weather_utils import
fetch_weather_forecast_summary, get_location_name
        from predict_yield import predict_yield
        from ai_features import explain_prediction

        weather_summary = None
        weather_times = []
        weather_temps = []
        weather_humidities = []
        weather_rains = []
        weather_location = ""
        yield_prediction = None
        explanation = "      ."

        if field.latitude and field.longitude:
            weather_data =
fetch_weather_forecast_summary(field.latitude,
field.longitude)
            weather_location =
```

Продовження лістингу А.1

```
get_location_name(field.latitude, field.longitude)

            if isinstance(weather_data, dict) and
"summary" in weather_data:
                weather_summary = weather_data["summary"]
                weather_times = weather_data.get("times",
[])
                weather_temps = weather_data.get("temps",
[])
                weather_humidities =
weather_data.get("humidities", [])
                weather_rains = weather_data.get("rains",
[])


                if field.harvest_status != "done":
                    try:
                        yield_prediction =
predict_yield(field, weather_summary)
                        explanation =
explain_prediction(yield_prediction, weather_summary)
                    except Exception as e:
                        explanation = f"   : {e}"

        return render_template(
            "field_detail.html",
            field=field,
            weather_summary=weather_summary,
            yield_prediction=yield_prediction,
            explanation=explanation,
            weather_times=weather_times,
            weather_temps=weather_temps,
            weather_humidities=weather_humidities,
            weather_rains=weather_rains,
```

Продовження лістингу А.1

```
            weather_location=weather_location
        )

    @app.route('/fields/history')
    @login_required
    def all_fields_history():
        history = FieldHistory.query \
            .join(Field) \
            .filter(Field.owner_id == current_user.id) \
            .order_by(Field.name,
FieldHistory.year.desc()) \
            .all()
        return render_template('all_fields_history.html',
history=history)

    @app.route('/field/<int:field_id>/edit',
methods=['GET', 'POST'])
    @login_required
    def edit_field(field_id):
        field = Field.query.get_or_404(field_id)
        if field.owner_id != current_user.id:
            flash("    .", "danger")
            return redirect(url_for('view_fields'))
        form = FieldForm(obj=field)
        if form.validate_on_submit():
            old_sowing_date = field.sowing_date
            old_crop = field.current_crop
            old_harvest_date = field.harvest_date

            field.name = form.name.data
            field.area = form.area.data
            field.soil_type = form.soil_type.data
            field.location = form.location.data
            field.current_crop = form.current_crop.data
```

Продовження лістингу А.1

```
            field.sowing_date = form.sowing_date.data
            field.latitude = form.latitude.data
            field.longitude = form.longitude.data
            field.harvest_status =
form.harvest_status.data
            field.harvest_date = form.harvest_date.data

            harvest_changed = field.harvest_date !=
old_harvest_date
            sowing_changed = field.sowing_date !=
old_sowing_date
            crop_changed = field.current_crop != old_crop

            should_add_to_history = (
                harvest_changed or
                ((sowing_changed or crop_changed) and
field.harvest_status == "")
            )
            if should_add_to_history:
                history = FieldHistory(
                    field_id=field.id,
                    crop_name=field.current_crop,
                    sowing_date=field.sowing_date,
                    harvest_date=field.harvest_date,
                    year=field.sowing_date.year if
field.sowing_date else datetime.now().year
                )
                db.session.add(history)
            db.session.commit()
            flash(" .", "success")
            return redirect(url_for('field_detail',
field_id=field.id))
        return render_template('edit_field.html',
form=form, field=field)
```

ДОДАТОК Б

Головний макет сайту

Лістинг Б.1 – Програмний код файлу base.html

```
<!doctype html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <title> </title>
  <link rel="stylesheet"
href="https://unpkg.com/leaflet@1.9.4/dist/leaflet.css" />
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/cs
s/bootstrap.min.css" rel="stylesheet">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.10.5/font/bootstrap-icons.css" rel="stylesheet">
  <link rel="stylesheet" href="{{ url_for('static',
filename='css/toast.css') }}">
  <link rel="stylesheet" href="{{ url_for('static',
filename='css/style.css') }}">
  <link rel="stylesheet" href="{{ url_for('static',
filename='css/loader.css') }}">
  <link rel="stylesheet" href="{{ url_for('static',
filename='css/employees.css') }}">
  <link rel="stylesheet" href="{{ url_for('static',
filename='css/view_employee.css') }}">
  <link rel="stylesheet" href="{{ url_for('static',
filename='css/welcome.css') }}">
  <style>
    html, body {
      height: 100%;
      min-height: 100vh;
      margin: 0;
```

Продовження лістингу Б.1

```
      padding: 0;
      background: url("/static/img/background.jpg") no-
repeat center center fixed;
      background-size: cover;
      background-attachment: fixed;
      background-repeat: no-repeat;
      background-position: center;
      font-family: "Segoe UI", sans-serif;
      overflow-x: hidden;
    }

    .glass {
       ...
      color: #222 !important;
     }
    body::after {
      content: '';
      position: fixed;
      top: 0;
      left: 0;
      width: 100%;
      height: 100%;
      background: rgba(0, 0, 0, 0.35);
      z-index: -1;
    }
    .content-wrapper {
      display: flex;
      justify-content: center;
      padding: 2rem;
      box-sizing: border-box;
      width: 100%;
    }
```

Продовження лістингу Б.1

```
  </style>

  <script
src="https://unpkg.com/leaflet@1.9.4/dist/leaflet.js"></sc
ript>
</head>
<body class="preload">

  <div id="loader-overlay">
    <div class="spinner-border text-light" role="status">
      <span class="visually-hidden">...</span>
    </div>
  </div>

  <div class="background-overlay"></div>

  <nav class="navbar navbar-expand-lg navbar-light bg-
light border-bottom shadow-sm mb-3">
    <div class="container-fluid">
      <a class="navbar-brand fw-bold" href="{{
url_for('index') }}">  </a>
      <button class="navbar-toggler" type="button" data-
bs-toggle="collapse" data-bs-target="#navbarNav">
        <span class="navbar-toggler-icon"></span>
      </button>

      <div class="collapse navbar-collapse"
id="navbarNav">
        <ul class="navbar-nav me-auto mb-2 mb-lg-0">
          {% if current_user.is_authenticated %}
            <li class="nav-item"><a class="nav-link"
href="{{ url_for('view_fields') }}"> </a></li>
            <li class="nav-item"><a class="nav-link"
href="{{ url_for('employees') }}"> </a></li>
```

Продовження лістингу Б.1

```
            <li class="nav-item"><a class="nav-link"
href="{{ url_for('machinery_list') }}"> </a></li>
            <li class="nav-item"><a class="nav-link"
href="{{ url_for('animals') }}"> </a></li>
            <li class="nav-item"><a class="nav-link"
href="{{ url_for('add_income') }}">  </a></li>
            <li class="nav-item"><a class="nav-link"
href="{{ url_for('add_expense') }}">  </a></li>
            <li class="nav-item"><a class="nav-link"
href="{{ url_for('all_fields_history') }}">  </a></li>
            <li class="nav-item"><a class="nav-link"
href="{{ url_for('report') }}"> </a></li>
          {% endif %}
        </ul>
        <ul class="navbar-nav">
          {% if current_user.is_authenticated %}
            <li class="nav-item">
              <a class="nav-link text-danger" href="{{
url_for('logout') }}"> </a>
            </li>
          {% else %}
            {% if request.path == '/login' %}
              <li class="nav-item">
                <a class="nav-link text-success" href="{{
url_for('register') }}"> </a>
              </li>
            {% elif request.path == '/register' %}
              <li class="nav-item">
                <a class="nav-link text-success" href="{{
url_for('login') }}"> </a>
              </li>
            {% endif %}
          {% endif %}
        </ul>
```

Продовження лістингу Б.1

```
      </div>
    </div>
  </nav>

  <div class="content-wrapper">
    {% block content %}{% endblock %}
  </div>

  <div class="toast-container position-fixed top-0 end-0
p-3" style="z-index: 1055;">
    {% with messages =
get_flashed_messages(with_categories=True) %}
      {% if messages %}
        {% for category, message in messages %}
          <div class="toast align-items-center text-bg-{{
category }} border-0 show mb-2" role="alert">
            <div class="d-flex">
              <div class="toast-body">
                {{ message }}
              </div>
              <button type="button" class="btn-close btn-
close-white me-2 m-auto" data-bs-dismiss="toast" aria-
label=""></button>
            </div>
          </div>
        {% endfor %}
      {% endif %}
    {% endwith %}
  </div>

  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/
bootstrap.bundle.min.js"></script>
  <script src="{{ url_for('static', filename='js/toast-
```

Продовження лістингу Б.1

```
init.js') }}"></script>
  <script src="{{ url_for('static', filename='js/map.js')
}}"></script>
  <script src="{{ url_for('static', filename='js/login-
effect.js') }}"></script>
  <script src="{{ url_for('static',
filename='js/background.js') }}"></script>
  <script src="{{ url_for('static', filename='js/map-
mini') }}"></script>
  <script src="{{ url_for('static', filename='js/preload-
bg.js') }}"></script>
  <script src="{{ url_for('static',
filename='js/employees.js') }}"></script>
  <script>
    window.addEventListener("load", () => {
      document.body.classList.remove("preload");
      document.body.classList.add("loaded");
      document.getElementById("loader-
overlay").style.display = "none";
    });
  </script>
{% include "chatbot_widget.html" %}
</body>
</html>
```

ДОДАТОК В

Інтерфейс вебсистеми

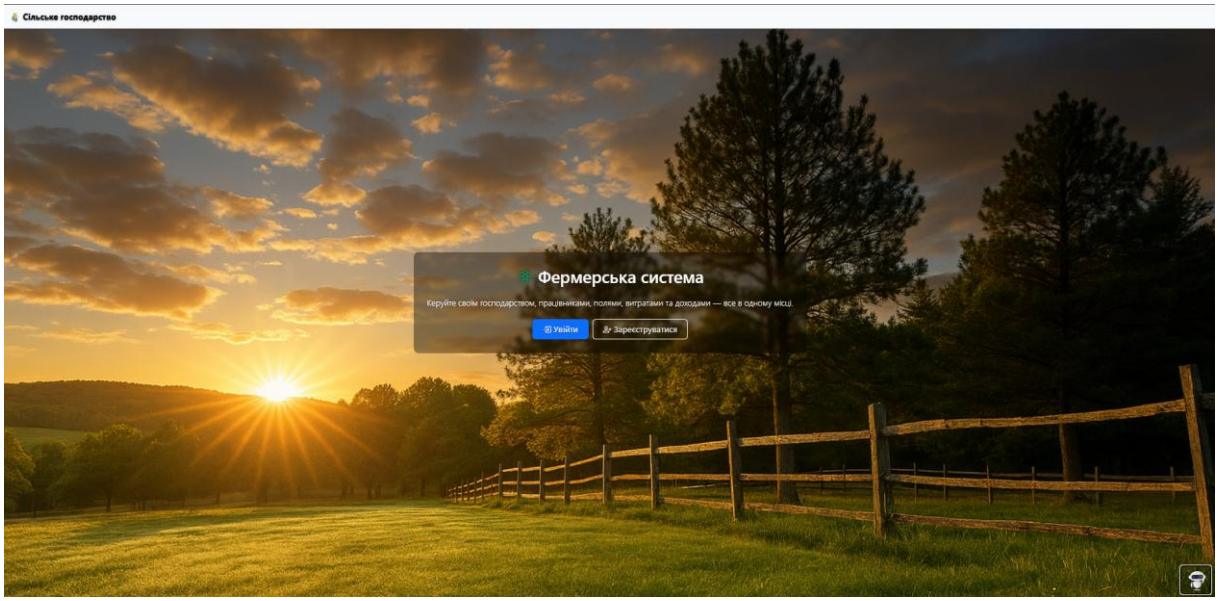

Рисунок В.1 – Головна сторінка системи до входу з кнопками входу та реєстрації

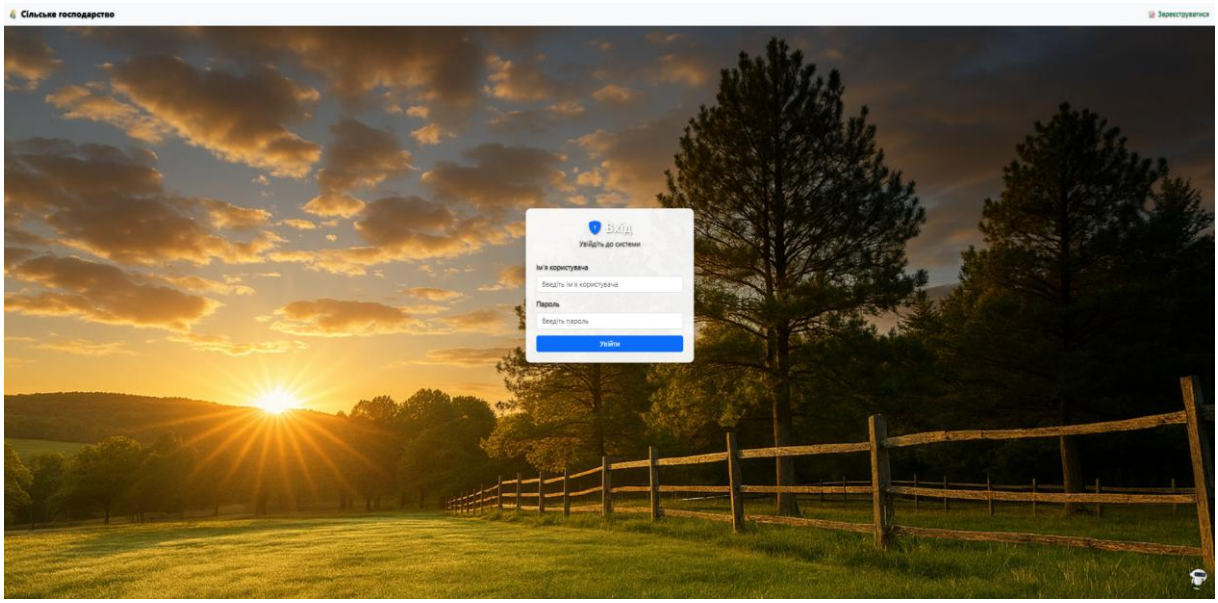

Рисунок В.2 – Форма входу до системи з полями для введення імені користувача та пароля

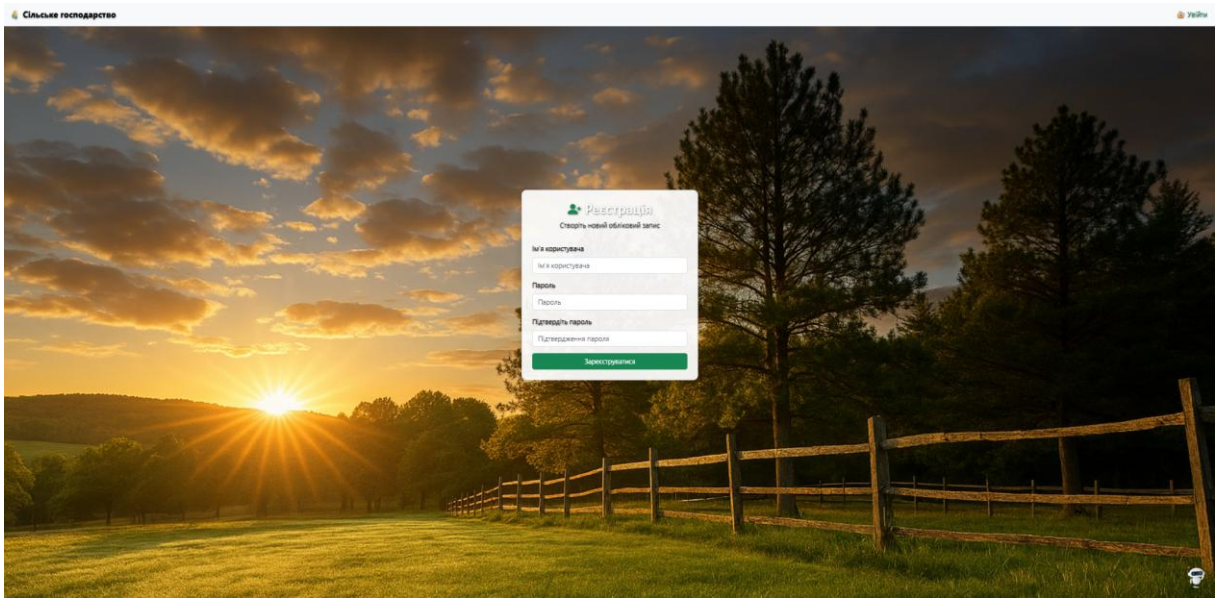

Рисунок В.3 – Форма реєстрації нового користувача у вебсистемі

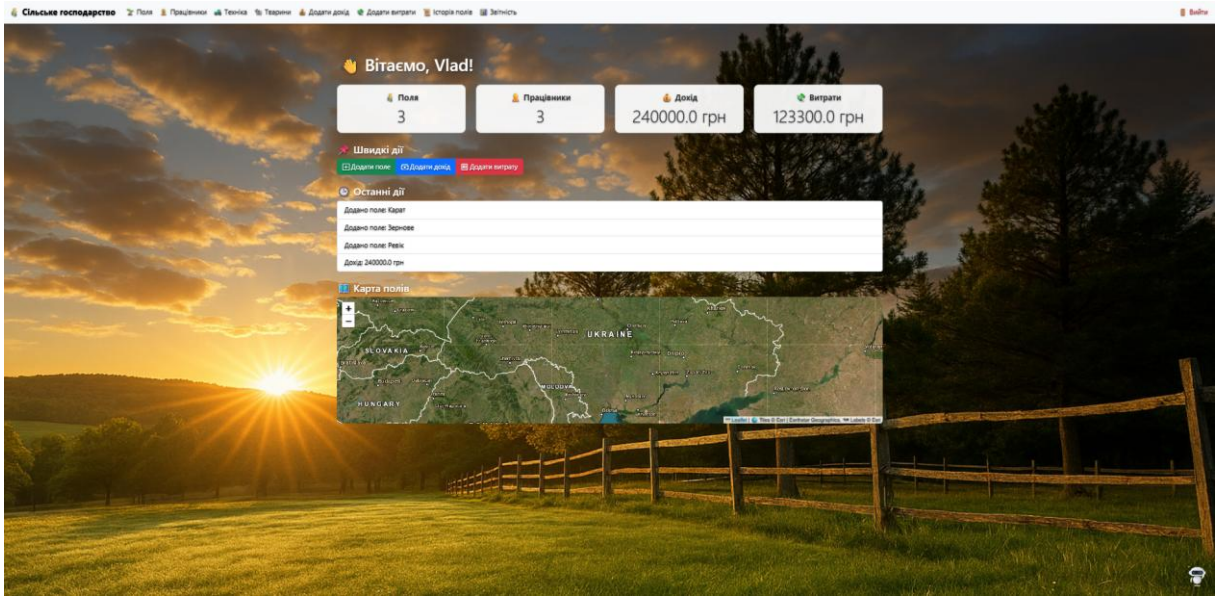

Рисунок В.4 – Головна інформаційна панель

Рисунок В.5 – Головна навігаційна панель до всіх основних розділів

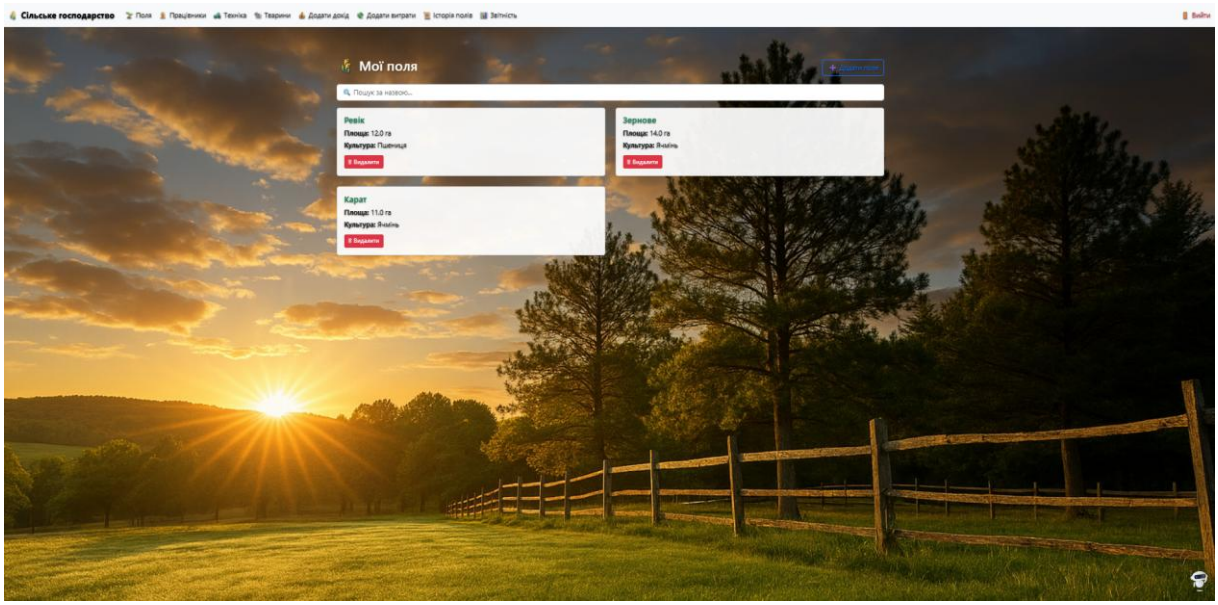

Рисунок В.6 – Інтерфейс обліку полів із можливістю пошуку, перегляду та видалення

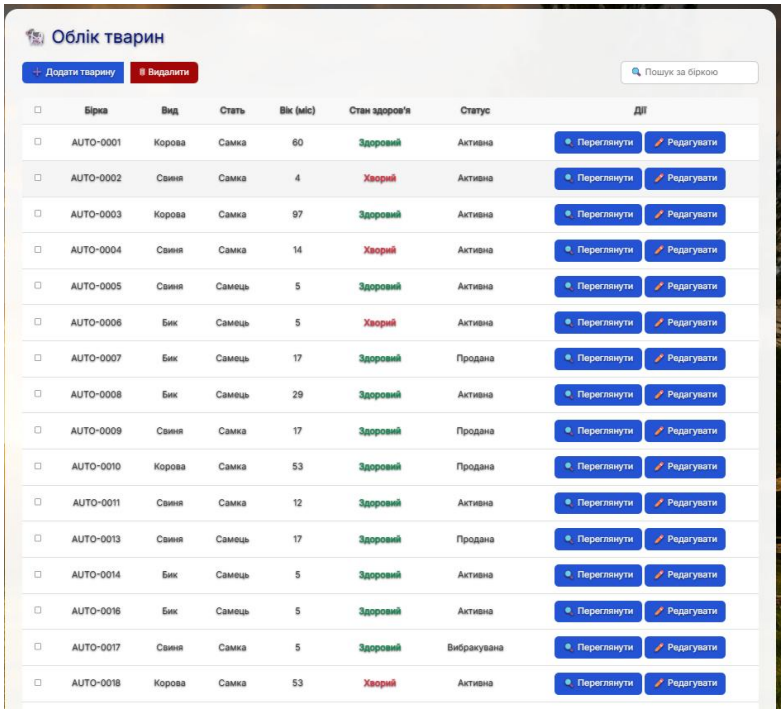

Рисунок В.7 – Інтерфейс обліку тварин

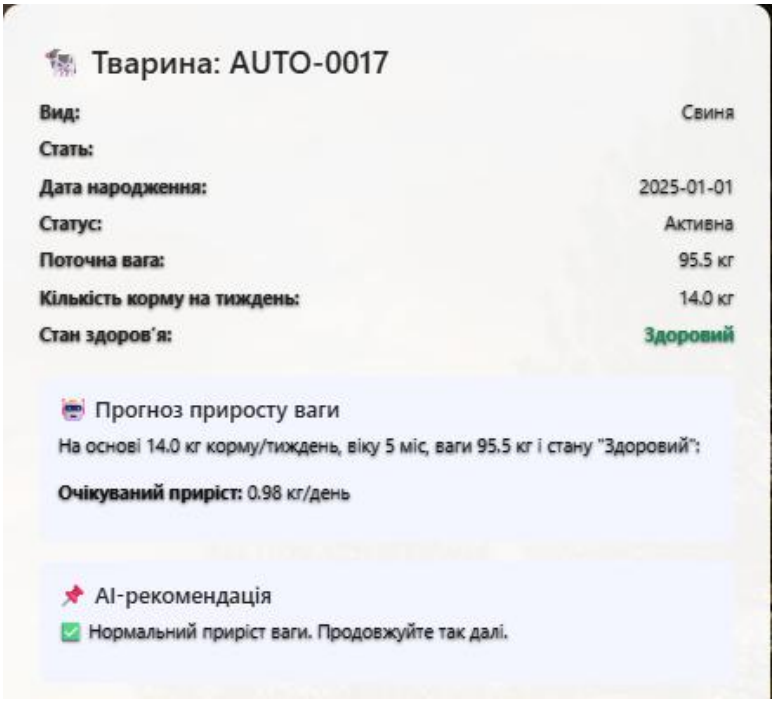

Рисунок В.8 – Детальна картка тварини з прогнозом приросту та AI-рекомендацію

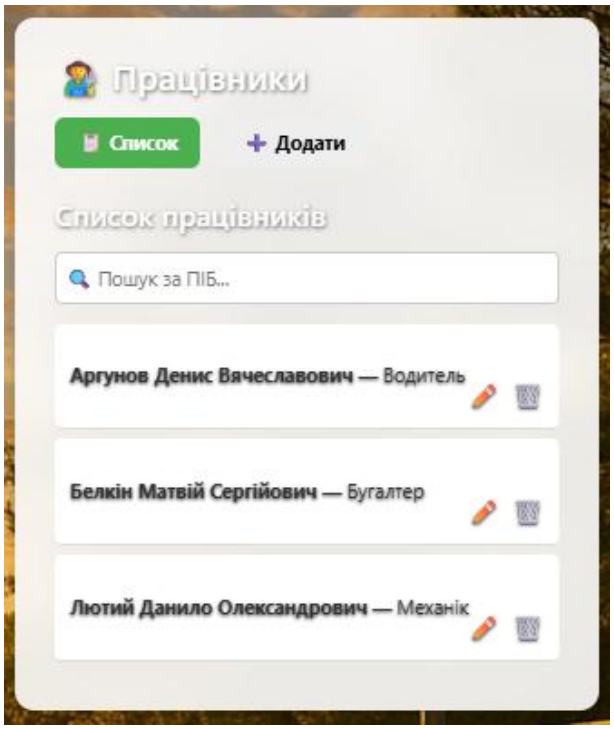

Рисунок В.9 – Інтерфейс модуля обліку працівників

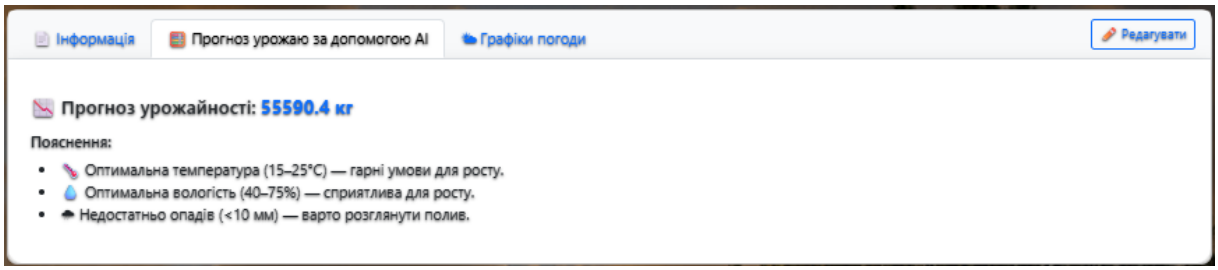

Рисунок В.10 – Інтерфейс прогнозу урожайності з поясненням впливу погодних умов

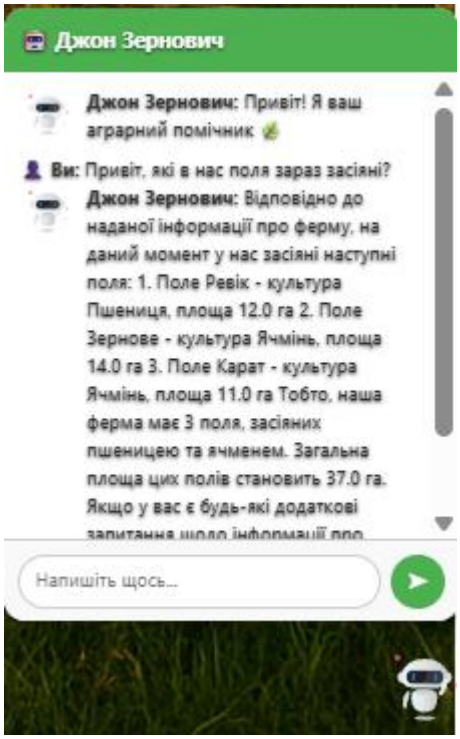

Рисунок В.11 – Вікно інтерактивного чат-бота ClaudeAI з відповіддю на запит

ДОДАТОК Г

Відомість кваліфікаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	89 с.
	Інші документи	
2.	Презентаційні матеріали	__ плакатів

.	.	Прізвище та ініціали.	Підп.	Дата			
Розробив		Путятин В.Є.			Вебсистема управління фермерським господарством з використанням штучного інтелекту	Шифр групи	Код напр./спец.
Перевірив		Павленко Є.П.				ІТШІ-21-2	122
Н.контр.		Малєєва І.А.				ХНУРЕ кафедра ШІ	
Затв.		Золотухін О.В.					