Skin Disease Detector

Project Overview

Skin Disease Detector is a Django-based web application designed to help users upload images of skin conditions and receive predictions about possible skin diseases using a machine learning model. The application supports user registration, login, profile management, and maintains a history of predictions for each user.

Features

- User authentication (registration, login, logout)
- Image upload for skin disease prediction
- Integration with Roboflow machine learning model for disease classification
- User profile management with profile picture upload
- Prediction history tracking
- Detailed disease information display

Setup and Installation

Prerequisites

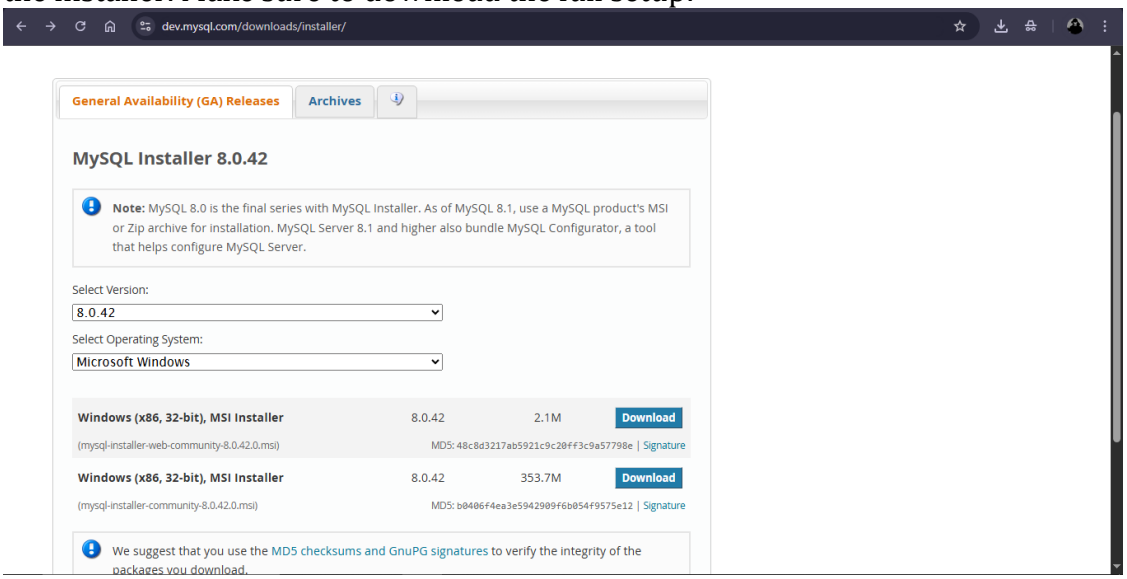
- Python 3.8+
- pip (Python package installer)
- Virtual environment tool (optional but recommended)

Cloning Repository

```
git clone https://github.com/arslan548/Skin_detector.git
cd Skin_detector
```

Database setup

1. Go to the MySql site (https://dev.mysql.com/downloads/installer/) and download the installer. Make sure to download the full setup:



2. Then install the MySql
3. You will create a username (By default its “root”) and a password during installation make sure to remember that.
4. After installation go to the folder “skindetector” in the repository you just cloned.
5. Now open settings.py in it and go to line 94, 95 and enter you usernaame and password.
6. Open terminal and give command:

```
mysql -u root -p
```

7. Enter your Mysql password.
8. Now enter this:

```
CREATE DATABASE skin_db;
```

9. Database setup complete.

Installation Steps

1. Create and activate a virtual environment:

   ```
   python -m venv venv
   .\venv\Scripts\activate  # On Windows: venv\Scripts\activate
   ```

2. Install dependencies:

   ```
   pip install -r requirements.txt
   ```

3. Apply database migrations:

   ```
   python manage.py migrate
   ```

4. Create a superuser (optional, for admin access):

   ```
   python manage.py createsuperuser
   ```

5. Run the development server:

   ```
   python manage.py runserver
   ```

6. Access the application at (`http://127.0.0.1:8000/`)

Usage

- Register a new user or log in with existing credentials.
- Upload an image of a skin condition via the upload page.
- View the prediction results with possible skin diseases and confidence scores.
- Access your profile to view and edit your information.
- Review your prediction history to see past uploads and results.
- Explore detailed information about various skin diseases within the app.

Project Structure

```
skin-disease-detector/
│
├── detection/                    # Main Django app for skin disease detection
│   ├── migrations/               # Database migrations
│   ├── static/                   # Static files (CSS, images, JS)
│   ├── templates/                # HTML templates
│   ├── templatetags/             # Custom template tags
│   ├── admin.py                  # Admin site configuration
│   ├── apps.py                   # App configuration
│   ├── disease_data.py           # Disease information data
│   ├── forms.py                  # Django forms for user input
│   ├── models.py                 # Database models
│   ├── signals.py                # Django signals for user profile management
│   ├── tests.py                  # Unit tests
│   ├── urls.py                   # App URL routes
│   └── views.py                  # View functions handling requests
│
├── media/                        # Uploaded media files (profile pics, images)
├── skindetector/                 # Project configuration
│   ├── __init__.py
│   ├── asgi.py
│   ├── settings.py               # Project settings and configuration
│   ├── urls.py                   # Project URL routes
│   └── wsgi.py
│
├── manage.py                     # Django management script
├── requirements.txt              # Python dependencies
├── README.md                     # Project documentation (this file)
└── roboflow_response.json        # Sample prediction response
```

Main Components

Models

- **Profile**: Extends the built-in User model with profile picture.
- **SkinImage**: Stores uploaded skin images.
- **SkinPrediction**: Stores prediction results linked to users.

Views

- User registration, login, logout
- Image upload and prediction
- Profile view and edit
- Prediction result display
- Disease detail information
- Prediction history

Forms

- User registration and login forms
- Profile update forms
- Image upload and prediction forms

Templates

Located in `detection/templates/detection/`, these HTML files render the UI for various pages such as login, register, profile, upload, prediction results, etc.

Static Files

Located in `detection/static/`, includes images and other static assets.

Running Tests

Run the Django test suite with:

```
python manage.py test
```

Extending the Project

- Add new disease data in `detection/disease_data.py`
- Enhance the machine learning model integration in `detection/views.py`
- Customize templates and static files for UI improvements
- Add more unit tests in `detection/tests.py`

Contributing

Contributions are welcome! Please fork the repository and submit pull requests for review.

License

This project is licensed under the MIT License - see the LICENSE file for details.

Contact

For questions or support, please contact the project maintainer at:
[arslansajjad548@gmail.com]

Additional Notes

- The project uses Roboflow API for skin disease prediction.
- Media files are stored in the `media/` directory.
- User authentication is handled by Django's built-in auth system.