Edmund Balnaves

23 Artificial Intelligence in Libraries on $5 per Day: Image Matching with Koha

Abstract: This chapter takes the reader on a journey through artificial intelligence (AI) integration with a well known open-source library management system, Koha. A case study explores the integration of an open-source AI toolkit with an open-source library management system. By leveraging open source and cloud services, the integration of an image recognition AI toolkit is demonstrated on a budget of under $5 per day. This chapter explores the strengths and opportunities in open-source AI projects and provides an example of integrating library and information services using a widely implemented open-source library management system.

Keywords: Open source software; Images; Library management systems (Computer systems); Koha

Introduction

The union of image capture from cameras with image recognition software has opened up new opportunities for many client-facing applications. When applied to libraries, the use of image recognition techniques to achieve image matching offers potential both in the semantic recognition of image and video assets held by the library, and in the enhancement of client-facing library applications.

This chapter explores the implementation of a facial recognition system using open-source toolkits when integrated with an open source library management solution. Open source refers specifically to projects – that have source code provided and allow royalty-free redistribution and modification of the code within the bounds of normal open-source licences (Balnaves 2012).

Many AI toolkits have their origins in open source, written either in Python or C++ programming languages. The use of open source allows inspection of the techniques used, and the nature of the community of developers allows an agile approach to development and testing of the fit of the tools developed to the tasks for which they are being employed. Most toolkits for AI have wrappers that allow them to be embedded using popular programming languages, including Python, Java and C++. Examples of AI open-source software include:

– OpenCV, a popular computer vision library that includes tools for face detection and recognition (OpenCV 2024)

https://doi.org/10.1515/9783111336435-024

- DLib C++ Library, a C++ library that includes Python bindings for facial landmark detection, face detection, and face recognition (DLibC++ Library 2022)
- Tensorflow, a machine learning framework that includes tools for face recognition (Tensorflow n.d.)
- Face_recognition, a simple Python face recognition library that wraps around dlib (Geitgey 2021)
- PyTorch, another Python machine learning framework that includes tools for face recognition (PyTorch 2024)
- Keras, a high-level deep learning application programming interface (API) that includes tools for face recognition (Keras n.d.)
- Apache MXNet, a deep learning framework that includes tools for face recognition. It is now retired and available in the attic (Mxnet 2022)
- Scikit-learn, a machine learning library that includes tools for facial recognition using support vector machine (SVM) and other algorithms (Scikit-learn 2024)
- Caffe, a deep learning framework that includes tools for face recognition (Caffe n.d.), and
- CodeProject AI, locally installed and self-hosted for any platform and any language. Runs as a Windows service or as a docker container (CodeProject AI 2024b).

Some of the software tools listed are used as building blocks for other software applications and make use of features from other applications.

An open-source implementation of AI tools has a range of privacy benefits to the library. Images and digital content are not sent to external services where the national hosting and privacy of content may be uncertain. One of the ethical considerations in the adoption of AI is the degree to which the algorithmic elements of the solution used can be scrutinised, tested and understood. In the case of open source, the library has agency over the algorithmic design of the AI implementation.

This chapter outlines a case study exploring the process of integrating an open-source AI toolkit with an open-source library management system. The process adopted explored the use of cloud services for storage to provide the proof-of-concept platform for the whole solution at low cost.

Choosing the Platform and Toolkit

Cloud platforms provide an environment that allow for low-cost proof of concept design of systems without having to invest in infrastructure in the first instance.

This case study chose the cloud infrastructure in Amazon Web services to demonstrate the use of its facilities and to illustrate the implementation of the proof-of-concept integration with the installation of a Koha library management system and the AI toolkit, using Koha plugins to bring the two solutions together. Such an approach allows free testing of solutions and services from an infrastructure perspective.

The implementation of the final solution can be done within the normal infrastructure of the institution. There are several advantages to implementing the initial trial solution within a cloud infrastructure. It allows the formulation and trialling of the final technical architecture required without having to commit to the architecture during the design stages. The solution sets can therefore be created and taken down quite easily while the project is in evolution. The cloud infrastructure allows experimentation with solution options without large upfront investment. The security architecture around the solution can be safely tested outside the target implementation environment.

The use of cloud platforms for initial proof of concept testing has many benefits. During the initial exploratory stage with software experimentation, the final target hardware environment may be unknown as different packages are tested for their suitability. In this case several open-source image recognition packages were explored before arriving at using the CodeProject AI toolkit. The CodeProject AI software was chosen because it:

- Installed easily
- Passed initial image testing well, and
- Provided a representational state transfer (REST) API for integration with other applications, which was the key requirement for the cross-platform integration in this case study.

Image Recognition Toolkit

Docker is a platform that allows lightweight installation of software based on containers that are pre-baked or prepared with the solution being installed. The beauty of docker is that it is platform-agnostic. Linux docker images can be installed on Windows systems for example, or Windows docker images can be installed on Linux systems, even though one is in open source, and one is proprietary to Microsoft. In the library and information science business, and in testing environments, docker images can be used to rapidly run in and test solutions. Docker images can also be used for some production solutions when thoroughly tested.

Docker lives on top of the host platform on which it is being run, and contains within its own system only the data which is sufficient for the operating system environment that is being used by the application in order to run the application. It is lightweight and because of this characteristic, it is easy to deploy. Docker provides an excellent method for undertaking software throw-away trials to test the suitability of software, and also for production deployment of software in scalable solutions.

For the purpose of the proof of concept in this case study, the image recognition toolkit sought required a good API and a web interface for testing purposes, and facial recognition features. Fortunately, in the AI space, there are many open-source solutions in both machine learning and image recognition some of which are listed earlier in this chapter.

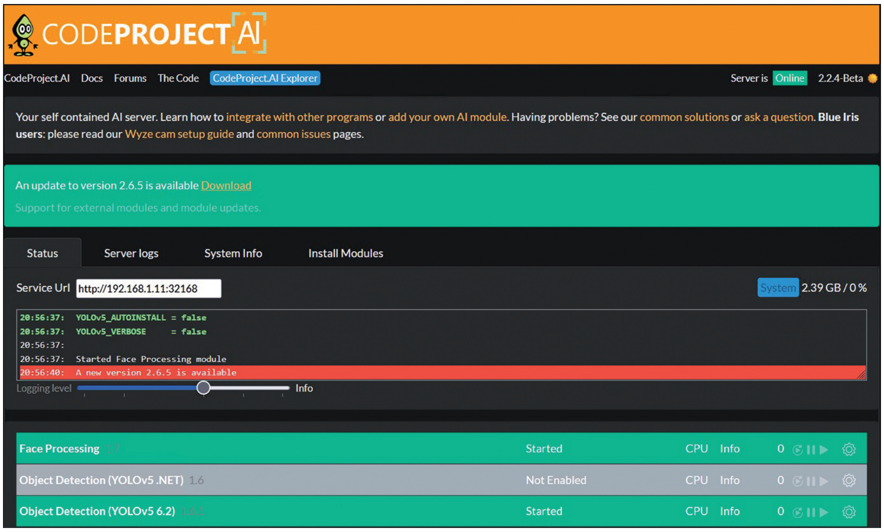
The case study undertook its exploration of integration using a kit recently released by CodeProject. "CodeProject is a community of Software Developers joined together with certain common goals: to learn, to teach and to have fun. Members from all over the world come together to share code, tutorials and knowledge for free to help their fellow Software Developers" (CodeProject 2024a). CodeProject makes available a range of solution kits for developers to experiment with new technologies. One of the recent solution toolkits is CodeProject AI, an implementation of Python open-source image recognition technologies within a Windows or Linux environment (CodeProject 2024b). The CodeProject AI solution has a docker install. There are many others, for example, FaceNet-Object-Detection-Net (Pandeyer 2024) and projects like Real Python (Stratis 2023).

The deployment of the CodeProject AI server was accomplished with a single command:

```
docker run -p 32168:32168 --name CodeProject.AI-Server -d -v /usr/share/
CodeProject/AI:/usr/share/CodeProject/AI codeproject/ai-server:1.6.8.0
```

Use of one command:
- Downloads the docker project
- Unpacks the docker images, and
- Launches the docker processes.

CodeProject AI makes its toolkit by default available via a web server and provides set up advice: "CodeProject.AI Server provides the glue, the infrastructure and the front-end to bundle together whatever AI project you wish to expose to your applications (Maunder 2024). They also kindly provide a web testing interface at a locally running API server with an internal address of http://localhost:32168/vision.html (Figure 23.1).


CODEPROJECT AI
CodeProject.AI Docs Forums The Code CodeProject.AI Explorer
Server is Online 2.2.4-Beta
Status Server logs System Info Install Modules

Your self contained AI server. Learn how to integrate with other programs or add your own AI module. Having problems? See our common solutions or ask a question. **Blue Iris users:** please read our Wyze cam setup guide and common issues pages.

An update to version 2.6.5 is available Download

Support for external modules and module updates.

Service Url http://192.168.1.11:32168 System 2.39 GB / 0 %

```
20:56:37:  YOLOv5_AUTOINSTALL = false
20:56:37:  YOLOv5_VERBOSE     = false
20:56:37:
20:56:37:  Started Face Processing module
20:56:40:  A new version 2.6.5 is available
```

Logging level Info

Face Processing 1.7	Started	CPU	Info	0
Object Detection (YOLOv5 .NET) 1.6	Not Enabled	CPU	Info	0
Object Detection (YOLOv5 6.2) 1.6.1	Started	CPU	Info	0

Figure 23.1: Web interface for CodeProject

The web interface (Figure 23.1) provided was used to trial some initial face recognition. The function allows upload of an image and testing recognition of image variants. The web interface has corresponding APIs which are the methods for integrating with other systems, in this case Koha.

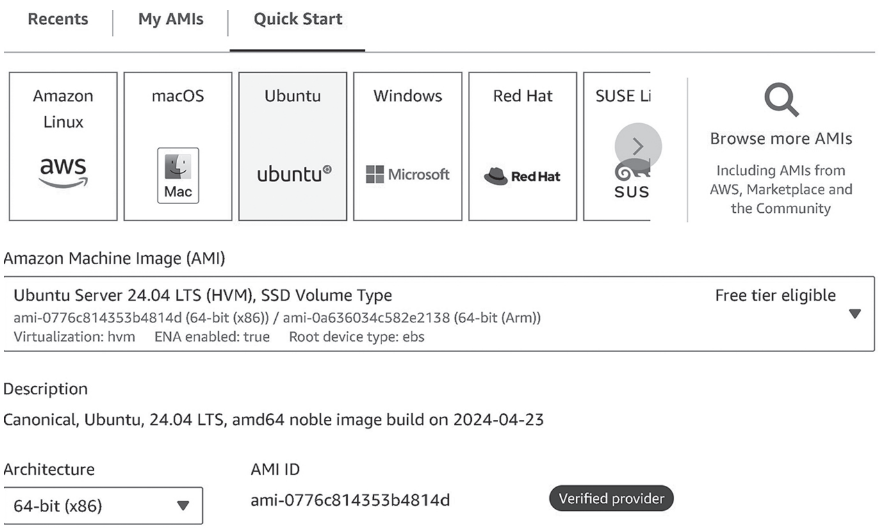
Cloud Server Setup

Cloud servers are available from many vendors and are increasingly positioned globally across most continents and larger industrial centres. A cloud server provides a platform which can be used to create and run computing resources on a fractional basis with micro-billing tied to the amount of resources used. For this reason, they provide a good environment for concept testing software architectures. A proof of concept of a new software solution can be tested in a an inexpensive throw-away environment without large upfront hardware expenditure. The final deployment may or may not be in a cloud platform.

This chapter focuses on the concept used to demonstrate the implementation of Koha version 21.11 on an AWS Ubuntu 20.04 server with integration to AI capabilities of CodeProject's AI docker images. It is easy to create an AWS server for throw-away testing. Creating the AWS instance requires that one:

- Registers an AWS account, free, but linked to a credit card, and
- Creates a server, EC2 in AWS terminology, and in this instance using the standard AWS Ubuntu 20.04 server open-source base. However, there are many flavours of computer operating systems that can be deployed at differing costs.

Koha and CodeProject were installed initially on a free tier eligible server following the standard installation guidelines for Koha on Debian (Koha 2024) using the Ubuntu 21.11 pathway (Figure 23.2).



Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-0776c814353b4814d (64-bit (x86)) / ami-0a636034c582e2138 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description
Canonical, Ubuntu, 24.04 LTS, amd64 noble image build on 2024-04-23

Architecture: 64-bit (x86)
AMI ID: ami-0776c814353b4814d
Verified provider

Figure 23.2: AWS Ubuntu server

In this project a small instance was created in the free tier layer. However, in the testing it was soon found that the image recognition software could not be installed in the smallest instance. The target installation required at least 4GB of memory. The problem was discovered when the server hung while launching the CodeProject docker image at the same time as the Koha server. The throw-away server just created was discarded and started again with a larger image. Final deployment took place on an 8GB (EC2 type "T3 large") server after creating and throwing away several server instances. The final cloud server had 2 x CPU and 8GB x memory. The path taken led successfully to a minimum architecture for the deployment environment, and to a minimal operational cost of $5 per day for the entire application operation including Koha.

Implementation of Koha

Koha is one of the most widely implemented library management systems in the world. It is an open-source library management solution supported by a rich community of developers, service providers and librarians around the globe. It runs on Debian, Ubuntu or other flavours of Linux and has a well-documented installation process (Koha 2024). Koha was installed on the EC2 instance alongside the docker CodeProject instance. It took twenty minutes to install and deploy Koha on the server with the advantage of familiarity.

Integrating with Koha

Koha provides a plugin system that allows integration of features within the application. The plugin system means that the developer does not have to touch the code base of the application. The plugin system itself has API hooks and hooks to place code within the application. In the case of this project, a plugin was developed based on existing community published plugins. The concept of plugins is common across many applications and allows extension of an application with local customisations without having to modify the underlying application. The development of this plugin was realised using readily available code examples for Javascript webcam integration, Koha plugin development and the CodeProject AI example integrations.

The plugin created achieved two minimal functions for the project integration task:

- An integration hook between the OPAC login page and the AI server to
  - Scan an image
  - Send the image to CodeProject AI, and
  - Authenticate the use on Koha based on a successful match, and
- An integration hook on the patron page to allow transfer of an uploaded patron image to CodeProject AI

Both development tasks essentially involved injecting JavaScript into the login page and librarian's patron editing page to upload images using the Koha Plugin API integration as follows (Figure 23.3):

- The OPAC login page shows JavaScript to capture an image using the webcam
- The JavaScript code sends the webcam image mage to the Koha server using the custom Koha Plugin integration with the Koha API, and

– The Koha plugin sends the image to the Code Project AI server for validation or registration.

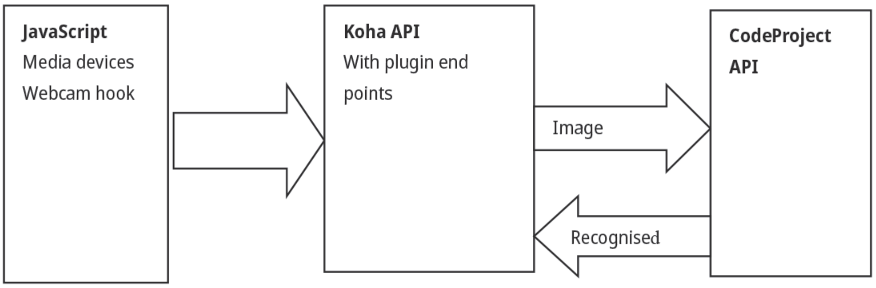


Figure 23.3: Koha integration

By relying heavily on existing plugin examples, and using the APIs in JavaScript, Koha and CodeProject, the proof of concept development took two days to complete and was the most complex programmatic task in the proof of concept. It involved software development focussed primarily on interacting with the Koha plugin functions and the CodeForLibAI functions.

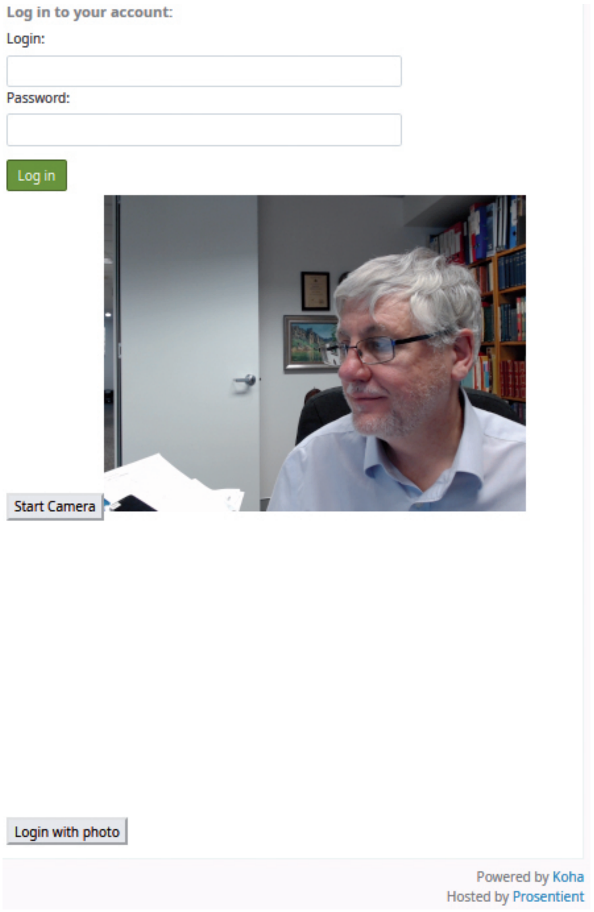


Figure 23.4: Online Public Access Catalogue (OPAC) login page

The online public access catalogue (OPAC) login page offers the user a choice in logging in and offers the use of the camera and image creation for sign in (Figure 23.4).

Testing the Toolkit

Artificial Intelligence systems rely on models of the real world of one sort or another. Image recognition for example can use many occurrences of sample images to train a model for subsequent recognition of similar occurrences. The larger the sample sizes available and the greater the amount of data included, the better the matching will be. When applying a real-world occasion against a model, the matching process is always an approximation: the image match is looking for equivalence or similarity in characteristics between the query image and the model and the resulting match is evaluated for performance. The match equals *this much* which is the level of confidence. The confidence level measures probability of accurate detection.

The matching confidence levels are crucial in the context of authenticating facial image recognition. If there is too low a confidence level accepted from the matching algorithm, any face is happily accepted in the authentication process. If an exceptionally high confidence level is reuired, it may be functionally impossible to log in. Initial explorations indicated a confidence level of > 0.75 as a minimum confidence.

An important aspect in using AI of any kind is the algorithm used and the methodology for use, and whether levels of confidence are provided at any of the modelling or processing layers. The benefit of using open source is that the whole process is open to scrutiny. The CodeProject AI yields a confidence level on its image and facial matching which can be used to good effect in the overall testing.

CodeProject AI does not store the original image once it has captured and processed the image. It simply stores a vector representing the face captured. Exploring the source reveals an SQLITE database. CodeProject itself is an application wrapper. It makes use of other open source projects to provide a simple interface to access a range of AI functions – which greatly simplifies the integration process with existing applications. CodeProject AI uses ObjectDetectionYolo (You Only Look Once) which itself uses OpenCV (Kundu 2023) and illustrates the ways in which open source allows progressively more complex solutions of tool kits from a wide collection base.

How does the toolkit store the facial image? No image is saved by the toolkit. Instead, it stores a vector defining the unique points defining the face, using the underlying image recognition software. It stores the vector in a database table, which can be explored (Figure 23.5). Note that this vector contains personally sensitive data as it provides defining facial recognition.

```
apt install SQLite
sqlite3 ./faceembedding.db
.tables
List the table
Select * from TB_EMBEDDINGS
```

Figure 23.5: Vector in database table

There is a set or a vector array that describes the unique points for a face (Figure 23.6).

```
0.026341065674227185, 0.03604118525981903, 0.05735778045518771,
-0.04988788140183555, -0.001674302271567285, 0.028918949887155787,
-0.006237712223018781, 0.008357822452184116, 0.08690883219222096,
0.03714309023022652, 0.03508520871400833, 0.016281576706784135,

......

0.057015251785516739, 0.008849352598190308, 0.06125432997941971,
-0.00867899414151907, 0.042281255453185, -0.010691865347325802,
-0.02570171467959881, -0.01565711200237274, -0.03459269553222928,
-0.011747458018362522, -0.03084883838891983, -0.05788338112831116,
-0.09049146135759354, -0.020388029515743256, -0.061122243144919586,
0.026703915790077972, 0.0673513263463974, 0.04039162260029793,
0.018179498612880707]
```

Figure 23.6: Example of a vector array for a face N.B. It does not represent a real face

The API call returns matched the facial identifications and produced the confidence level for the match. The application using the toolkit is left to make a decision on what to do based on the confidence levels of the match.

Webcam Integration, Secure Server and Image Scanning

To be useful within the Koha application, it is necessary to allow Koha to take an image snapshot to send to the server. Accessing the webcam from the browser to scan facial images might have been the trickiest part of the project:. Fortunately, the problem was solved several years ago with the browser introduction of Hypertext Mark Up Language 5 (HTML5) and media functions for the browser which allow JavaScript functions to access and manipulate images from the webcam.

One of the first integration issues with the web interface for Koha concerned accessing the webcam. For good security reasons, one cannot access the webcam from a non-SSL website. The reason for this is sound: anyone on the local network could watch or listen to a video or audio stream if the stream were not sent through a secure connection. Firefox and Chrome disable access to webcam functions if the web page itself is not secure.

Secure socket layer (SSL) is the technology used to create a secure information interchange between the browser being used whether it be Chrome, Firefox or some other, and the server to which one is connecting. Websites use public certificates shared with the browser during a search to create a secure connection.

Creating an SSL (https) protected website is an additional step required for proof of concept for the project being undertaken. While there are options for what are called self-signed certificates, they do not always overcome the obstacle of using secured components of the browser. To create an SSL-protected website there are two components:

– Creating a domain name, which will be the public web domain of the website on which Koha will be hosted. The domain name chosen was aidemo.intersearch.com.au. Creating a domain name requires the use of a domain registration company like GODADDY or one's own organisation if it has the capability, and
– Installing a secure certificate for the domain. An SSL certificate needs to be installed on the server being used. The certificate is time limited and is currently one year and must be created through a well-known registration system that is widely recognised on client computers and devices. There may be a fee associated with the certificate creation, but some sites provide the certificate for free as part of the hosting.

The project described in this case study had control of an existing domain, intersearch.com.au, and two sub-domains for Koha were registered: aidemo.intersearch.com.au and aidemoadmin.intersearch.com.au. The two domains translated to the cloud web server and the following domains were created, and certificates registered for them: https://aidemo.intersearch.com.au, and https://aidemoadmin.intersearch.com.au.

Fortunately, there is a free certificate service called Let's Encrypt (https://letsencrypt.org/) which provides an open-access method for registering certificates at no charge. The excellent service provides a solid and robust means for encryption of a website and its existence and activities have been responsible for a much greater level of systematic website encryption. Let's Encrypt is a non-profit authority which accepts sponsorships and donations. Support for this wonderful service is greatly to be encouraged. Let's Encrypt's products can be integrated with the servers very easily with a tool certbot (Inmotion Hosting Support Center 2023). It takes three steps in Linux:

– *Install snapd:*
  – sudo apt install snapd
  – sudo snap install core; sudo snap refresh core

- *Install Certbot with snapd:*
  - sudo snap install --classic certbot
- *Create a symlink to ensure Certbot runs:*
  - sudo ln -s /snap/bin/certbot /usr/bin/certbot

Once installed, the certbot tool can be used to create certificates that are automatically renewed on the server. With encryption enabled on the web server, the integration with the computer webcam followed naturally.

Ethical Review

Any project that captures biometric data, or data that might be interpreted as biometric, raises ethical considerations in relation to retention, access, appropriate use and consent:

> **Retention:** For how long is the image data to be retained? Will the image data be removed after its relevant usage or when the client is no longer a library subscriber? While the AI image processor does not retain the original image, the vector definition of the face in itself constitutes sensitive personal data and should not be retained once the client is no longer a member of the library. The client must be able to remove all representations of personal biometric data. Koha provides this capability.
>
> **Access:** Who can access the library subscriber data, especially where it contains biometric information such as the patron image? Are there security controls to access, or example two factor authentication).
>
> **Use:** Is the captured image data used appropriately and only for designated permitted purposes? In this case its purpose was to provide a simple, hands free, face-based login. There are obvious weaknesses with such a system that might not make it suitable in, for example, a public library setting. It might, however, be very useful in a 24 x 7 professional or research library setting.
>
> **Consent:** The use of the face recognition login should not be activated automatically. In this case it was triggered by the user, should s/he wish to use this method for authentication. Traditional password authentication was enabled during the proof of concept to allow flexible opt in/opt out for the new authentication path.

Concluding the Proof of Concept and Moving to Production

The purpose of the exploration described in this case study was to determine the feasibility and complexity of integrating an image recognition toolkit with the well-known open-source library system Koha. The journey demonstrated that such a toolkit could be constructed using a range of components and methodologies with minimal infrastructure costs, indeed on a budget of $5 per day.

The next stage of the journey is a lengthy process. Moving from proof of concept to production and implementation requires considerably more testing, an evaluation of scalability and a review of security and ethical issues. The toolkit chosen and others should be tested against the same methodology to evaluate the most effective method for achieving the desired outcomes. The same kind of agile proof of concept can help to quickly determine the viability of the approach and which of the possible solutions might represent an appropriate path forward.

Following further testing of the methods used, the considerations for production implementation include:

- Testing across different devices and environments
- Establishing scalability of the solution – what happens when 1000 people are using it?
- Testing across different facial profiles, population, and gender sets
- Conducting a security review of the solution. How safe are all the components used? What vulnerabilities apply, and can they be mitigated? and
- Running a project ethical review. What are the implications for use in specific contexts?

The purpose of this exercise was to demonstrate the facility with which an agile process with open-source software components can be used to develop and test an AI application with very low cost. The project explored the use of AI integration methodologies with an existing open-source library management system. The many open-source AI toolkits available, and API extensions of commercial systems such as ChatGPT mean that system integration tasks with AI systems are not intrinsically technically complex to undertake.

References

Balnaves, Edmund. 2008. “Open Source Library Management Systems: A Multidimensional Evaluation.” *Australian Academic and Research Libraries* 39, no. 1: 1–13. https://doi.org/10.1080/00048623.2008.10721320.

Caffe. n.d. “Deep Learning Framework by BAIR.” https://caffe.berkeleyvisiGeiscilit.org.

CodeProject. 2024a. “CodeProject: A Guide: What is CodeProject?” https://www.codeproject.com/info/guide.aspx.

CodeProject. 2024b. “CodeProject.AI Server: AI the Easy Way.” February 29, 2024. https://www.codeproject.com/Articles/5322557/CodeProject-AI-Server-AI-the-easy-way.

Dlib C++ Library. 2022. http://dlib.net/.

Geitgey, Adam. 2021. “Face_recognition.” v1.2.2 latest April 3, 2018. https://github.com/ageitgey/face_recognition.

Inmotion Hosting Support Center. 2023. “How to Install Let’s Encrypt SSL on Ubuntu with Certbot.” Updated December 1, 2023. https://www.inmotionhosting.com/support/website/ssl/lets-encrypt-ssl-ubuntu-with-certbot/.

Keras. n.d. https://keras.io/.

Koha. 2024. “Koha on Debian.” Last modified March 18, 2024. https://wiki.koha-community.org/wiki/Koha_on_Debian.

Kundu, Rohit. 2023. “ YOLO: Algorithm for Object Detection Explained [+Examples]”. *V7Labs: Blog.* January 17, 2023. https://www.v7labs.com/blog/yolo-object-detection.

Maunder, Chris. 2024. “CodeProject.AI Module Creation: A Full Walkthrough in Python.” February 15, 2024. https://www.codeproject.com/Articles/5377531/CodeProject-AI-Module-creation-A-full-walkthrough.

Mxnet. 2022. “Apache Mxnet: A Flexible and Efficient Library for Deep Learning.” https://mxnet.apache.org/versions/1.9.1/.

OpenCV. 2024. “Open CV University.” https://opencv.org.

Pandeyer, Yash. 2024. “YashNita /FaceNet-Object-Detection-Net-.” https://github.com/YashNita/FaceNet-Object-Detection-Net-?tab=readme-ov-file#readme.

PyTorch. 2024. https://pytorch.org/.

Scikit-learn. 2024. “Machine Learning in Python.” https://scikit-learn.org/stable/index.html.

Stratis, Kyle. 2023. “Build Your Own Face Recognition Tool with Python.” *Real Python.* April 24, 2023. https://realpython.com/face-recognition-with-python/

Tensorflow. n.d. “An End-to-end Platform for Machine Learning. TF2.16 Released.” https://www.tensorflow.org/.