

HEAVENS' LIGHT IS OUR GUIDE



RAJSHAHI UNIVERSITY OF ENGINEERING & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE &
ENGINEERING

CSE 3100
WEB-BASED APPLICATION PROJECT

HCV-Ai: Non-invasive Hepatitis C Virus Detection System

Submitted by

Kaif Ahmed Khan

Roll: 2103163

Submitted to

Md. Farhan Shakib

Lecturer

August 27, 2025

Contents

1	Introduction	1
2	System Components	1
2.1	Frontend Components	1
2.2	Backend Components	1
2.3	Dataflow Diagram	1
2.4	User Interaction Diagram	1
2.5	Key Features	1
3	Tools & Technologies	6
	Next.js	7
	FastAPI	7
	SQLite	7
	Scikit-Learn	7
	NGINX	7
	Docker	7
4	Sustainability & Environmental Considerations	9
	Performance Optimization	9
	Energy Efficiency	9
	Long-Term Impact	9
5	Project Management & Cost Estimation	10
6	Conclusion	10

List of Figures

1	Dataflow diagram of the website	1
2	Sequence diagram for showing how the user interacts with the website	2
3	Prediction system and prediction history in dashboard	3
4	User Management	4
5	FastAPI backend documentation @ https://172.104.62.100/api/docs	5
6	Docker running on linode ubuntu server vps	8

1 Introduction

The web application brings to life a machine learning framework developed for the non-invasive detection of Hepatitis C Virus (HCV), as outlined in our recent research. Designed with accessibility and accuracy in mind, the platform allows users—particularly patients and healthcare providers—to upload standard laboratory data and receive predictions about the possible stage of HCV infection: healthy, hepatitis, fibrosis, or cirrhosis.

The primary objective of the application is to make early and accurate HCV detection more accessible, especially in low-resource settings where traditional diagnostic methods are often invasive, costly, and difficult to access. By leveraging routine blood test results and state-of-the-art machine learning techniques—including synthetic data generation, feature selection, and explainable AI—this application aims to provide reliable, interpretable predictions that can assist in timely clinical decision-making.

This system addresses a critical gap in healthcare: the lack of affordable and scalable tools for multiclass liver disease classification. Whether you’re a clinician, researcher, or concerned individual, this platform empowers you with actionable insights using only non-invasive lab data.

2 System Components

2.1 Frontend Components

2.2 Backend Components

2.3 Dataflow Diagram

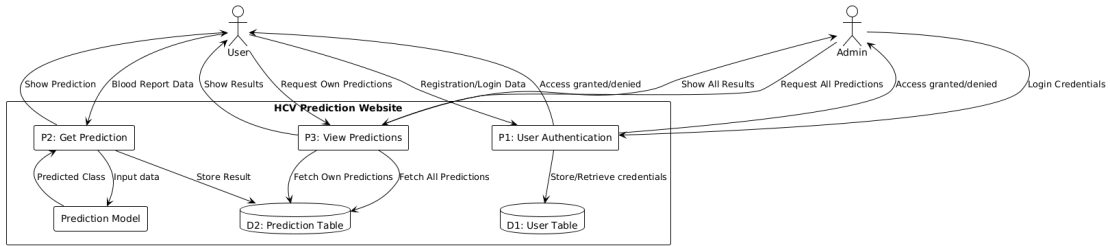


Figure 1: Dataflow diagram of the website

2.4 User Interaction Diagram

2.5 Key Features

User Management

- User registration and login system with SQLite database authentication.
- Role-based access control: Patients (users) and Admins.
- Secure storage of user information.

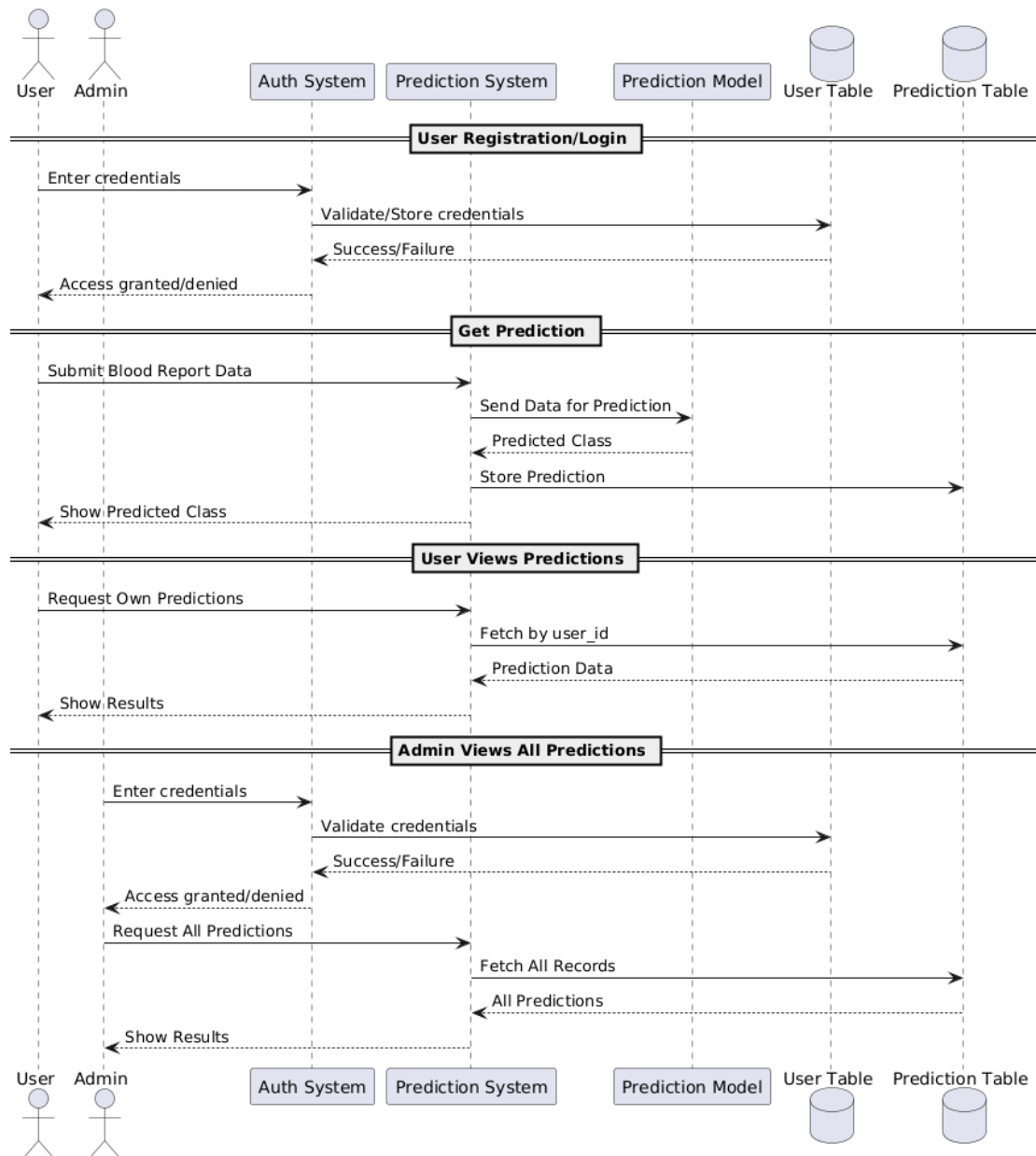


Figure 2: Sequence diagram for showing how the user interacts with the website

HCV Predictor

Dashboard

Predict

Profile

Logout

HCV Risk Prediction

Required Parameters

ALB (Albumin) *

e.g., 47

ALP (Alkaline Phosphatase) *

e.g., 37.9

AST (Aspartate Aminotransferase) *

e.g., 7.1

CHE (Cholinesterase) *

e.g., 6.6

CGT (γ-glutamyltransferase) *

e.g., 12.1

Optional Parameters

CREA (Creatinine)

0

CHOL (Cholesterol)

0

PROT (Protein)

0

BIL (Bilirubin)

0

ALT (Alanine Aminotransferase)

0

Age

0

Sex

Male

Get Prediction

About HCV Prediction

This tool uses machine learning to predict the risk of Hepatitis C Virus (HCV) infection based on blood test parameters. It is intended for informational purposes only and should not replace professional medical advice. Always consult a healthcare provider for diagnosis and treatment.

Developed by

Kaif Ahmed Khan

HCV Predictor

Dashboard

Predict

Profile

Logout

HCV Risk Prediction

Required Parameters

ALB (Albumin) *

47

ALP (Alkaline Phosphatase) *

37.9

AST (Aspartate Aminotransferase) *

48.4

CHE (Cholinesterase) *

10.3

CGT (γ-glutamyltransferase) *

68.2

Optional Parameters

CREA (Creatinine)

0

CHOL (Cholesterol)

0

PROT (Protein)

0

BIL (Bilirubin)

0

ALT (Alanine Aminotransferase)

0

Age

0

Sex

Male

Get Prediction

Prediction Results

Prediction ID:

4

HCV Status:

Hepatitis

Risk Level:

Medium

Date:

8/27/2025, 6:37:01 PM

Hepatitis Detected

The results indicate hepatitis. Please consult with a healthcare professional immediately for proper diagnosis and treatment plan.

Test Parameters

ALB: 47

ALP: 37.9

AST: 48.4

CHE: 10.3

CGT: 68.2

New Prediction

Print Results

Developed by

Kaif Ahmed Khan

(a) Prediction Page

(b) Showing prediction result

HCV Predictor

Dashboard



Predict

Profile

Logout

Prediction History

+ New Prediction

ID	DATE	RESULT	RISK LEVEL	ACTIONS
#1	8/28/2025	Hepatitis	Medium	 

(c) Prediction history in dashboard

Prediction Details

Prediction ID

#1

Date

8/28/2025, 3:31:39 PM

Result

Hepatitis

Risk Level

Medium

Clinical Assessment

Hepatitis detected. Immediate medical consultation recommended.

Test Parameters

REQUIRED	OPTIONAL	DEMOGRAPHICS
ALB: 47	CREA: 0	Age: N/A
ALP: 37.9	CHOL: 0	Sex: Male
AST: 48.4	PROT: 0	
CHE: 10.3	BIL: 0	
CGT: 68.2	ALT: 0	

Parameter Definitions

ALB: Albumin

ALP: Alkaline Phosphatase

AST: Aspartate Aminotransferase

CHE: Cholinesterase

CGT: γ-glutamyltransferase

ALT: Alanine Aminotransferase

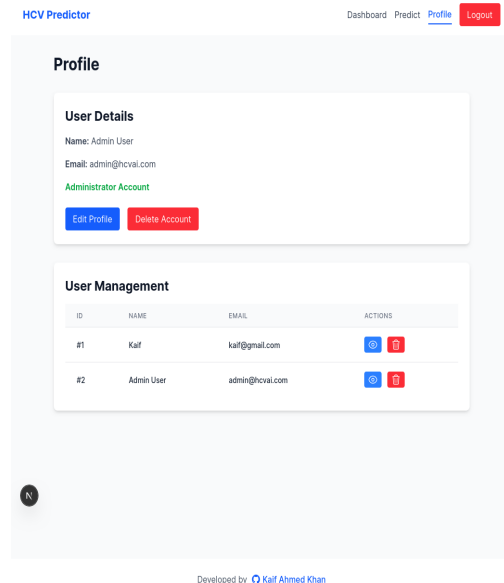
Print Report

Close

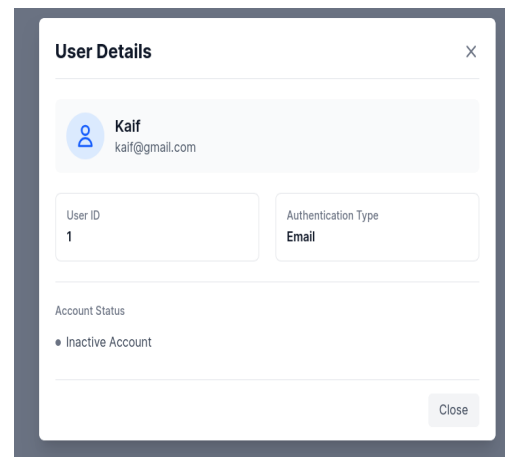
3

(d) Prediction details modal

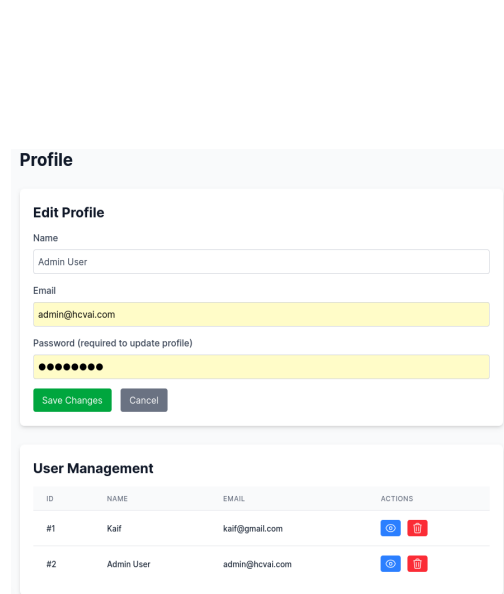
Figure 3: Prediction system and prediction history in dashboard



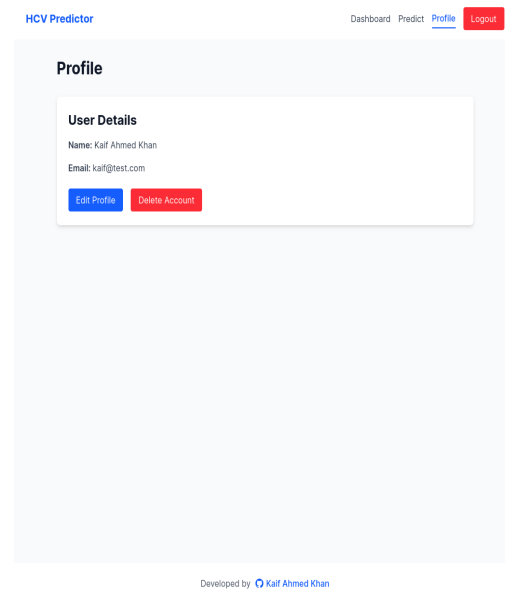
(a) Admin profile page



(b) User details modal



(c) Edit profile details



(d) Non-admin profile page

Figure 4: User Management

HCV AI Backend 1.0.0 OAS 3.1

/api/openapi.json

API for HCV AI application

Servers

/api

Authorize

Users

GET

/users/me

Get Me

PUT

/users/me

Update Me

DELETE

/users/me

Delete Me

GET

/users/{user_id}

Get User

GET

/users/

List Users

Auth

POST

/auth/register

Register

POST

/auth/login

Login

GET

/auth/google-login

Google Login

GET

/auth/callback

Google Callback

Predictions

GET

/predictions/

Get My Predictions

POST

/predictions/

Create Prediction

GET

/predictions/{prediction_id}

Get Prediction

DELETE

/predictions/{prediction_id}

Delete Prediction

GET

/predictions/user/{user_id}

Get Predictions For User

default

GET

/

Read Root

GET

/healthz

Check Api Health

Schemas

Body_login_auth_login_post > Expand all object

HTTPValidationError > Expand all object

PredictionCreate > Expand all object

PredictionOut > Expand all object

UserCreate > Expand all object

UserOut > Expand all object

ValidationError > Expand all object

Figure 5: FastAPI backend documentation @ <https://172.104.62.100/api/docs>

Prediction System

- Input form for blood report data (patient submits laboratory values).
- Pretrained machine learning model processes the input.
- Prediction of HCV stage (4 classes: 0–3).
- Prediction result is displayed instantly to the user.

Prediction History

- Patients can view their own previous predictions.
- Admins can view all users' predictions for monitoring and analysis.

Database Integration

- SQLite database with two main tables:
 - **Users Table:** Stores user credentials and profile information.
 - **Predictions Table:** Stores patient input, prediction results, and timestamps.

Security and Roles

- Authentication system for both patients and admins.
- Role-specific permissions (patients see only their own data, admins see all data).

3 Tools & Technologies

Languages

- HTML
- CSS
- JavaScript
- Python
- SQLite3

Next.js

FastAPI

SQLite

Scikit-Learn

NGINX

```
server {
    listen 80;
    server_name _;
    return 301 https://$host$request_uri;
}

server {
    listen 443 ssl;
    server_name _;

    ssl_certificate      /etc/ssl/cert.pem;
    ssl_certificate_key  /etc/ssl/privkey.pem;
    # Proxy Next.js frontend
    location / {
        proxy_pass http://frontend:3000;
        proxy_set_header    Host $host;
        proxy_set_header    X-Real-IP $remote_addr;
        proxy_set_header    X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header    X-Forwarded-Proto $scheme;
    }
    # Proxy FastAPI backend
    location /api/ {
        proxy_pass http://backend:8000/;
        proxy_set_header    Host $host;
        proxy_set_header    X-Real-IP $remote_addr;
        proxy_set_header    X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header    X-Forwarded-Proto $scheme;
    }
}
```

Docker

Justification for Choice of Technology Stack

The technology stack for the medical prediction website was carefully chosen to balance performance, scalability, and ease of development. Next.js was selected for the frontend because it provides a modern React framework with support for server-side rendering and static site generation, ensuring fast page loads and a responsive user interface.

FastAPI¹ was chosen for the backend due to its lightweight design, asynchronous capabilities, and automatic API documentation, which allows for efficient handling of requests and easy integration with machine learning models.

¹<https://fastapi.tiangolo.com/>

```
root@localhost:~# neofetch ascii --ascii_distro ubuntu_small --color_blocks off
root@localhost
-----
  OS: Ubuntu 24.04.3 LTS x86_64
  Host: Compute Instance
  Kernel: 6.8.0-79-generic
  Uptime: 2 hours, 58 mins
  Packages: 797 (dpkg)
  Shell: bash 5.2.21
  Resolution: 1280x800
  Terminal: /dev/pts/0
  CPU: AMD EPYC 7713 (4) @ 1.999GHz
  GPU: 00:01.0 Vendor 1234 Device 1111
  Memory: 384MiB / 7941MiB

root@localhost:~# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
f0ac4e332dab   hcv-ai-backend "uvicorn src.main:ap..." 2 hours ago   Up 2 hours   0.0.0.0:8000->8000/tcp, [::]:8000->8000/tcp   hcv-ai-backend-1

root@localhost:~# docker logs --tail 5 hcv-ai-backend-1
INFO: 167.94.138.190:36556 - "GET /favicon.ico HTTP/1.1" 404 Not Found
WARNING: Invalid HTTP request received.
WARNING: Invalid HTTP request received.
INFO: 167.94.138.190:41684 - "GET /favicon.ico HTTP/1.1" 404 Not Found
INFO: 167.94.138.190:41696 - "GET / HTTP/1.1" 200 OK
root@localhost:~#
```

Figure 6: Docker running on linode ubuntu server vps

SQLite serves as the database system because it is lightweight, file-based, and easy to set up without requiring complex configuration, making it well-suited for rapid development and deployment in small to medium-scale applications.

Finally, Scikit-Learn was used for the machine learning component due to its simplicity, extensive functionality for classification tasks, and seamless integration with Python, enabling quick development and deployment of the pretrained HCV prediction model. This combination of technologies ensures that the system is easy to develop, performant, and maintainable, while also allowing for future migration to more scalable databases if the application grows.

4 Sustainability & Environmental Considerations

In developing the medical prediction website, sustainability and environmental impact have been considered in both design and deployment. The following strategies are applied:

Performance Optimization

The backend, implemented using FastAPI, is lightweight and asynchronous, which reduces server load and ensures efficient handling of requests. The frontend, built with Next.js, supports static site generation and server-side rendering, which decreases repeated computation and improves response times. These optimizations reduce unnecessary resource consumption and improve the overall energy efficiency of the system.

Energy Efficiency in Coding & Infrastructure

The codebase follows modular and efficient practices to minimize redundant operations. Database queries are optimized to avoid excessive computation in SQLite. Containerization with Docker allows for consistent and isolated environments, reducing wasted resources across deployments. Additionally, horizontal scaling strategies are preferred over overprovisioning, ensuring resources are used only when required.

Long-Term Impact

By deploying the application on modern cloud platforms such as Render² the system benefits from shared, energy-efficient infrastructure. This reduces the need for dedicated high-power servers. The chosen stack (FastAPI+Next.js) supports scalability, which means the application can grow without significant increases in energy use per user. In the long term, the platform aims to provide medical predictions with minimal environmental footprint while supporting sustainable digital health practices.

²<https://hcv-ai.onrender.com/>

5 Project Management & Cost Estimation

Project Timeline

The development phase is designed to be lightweight and completed in approximately 6 weeks:

- Requirement Analysis and Design: 0.5 week
- Backend Development (FastAPI): 1 week
- Frontend Development (Next.js): 1 week
- Database Setup (SQLite): 0.5 week
- Integration and Dockerization: 1 week
- Testing and Debugging: 1 week
- Deployment and Documentation: 1 week

Cost Estimation

The project is planned with minimal cost by leveraging free and open-source technologies:

- **Development Effort:** The project is developed by a single developer as part of an academic exercise, so direct labor cost is negligible. If monetized, the equivalent effort can be valued at approximately \$500–\$800 (including the developer hiring cost).
- **Operational Costs:** VPS like Linode or AWS EC2 with minimal hardware configuration cost upto \$5-\$10 per month (for backend and database) and Render (hosting) are sufficient for deployment. Docker is open-source and free. A custom domain is optional, costing about \$10 per year.

Overall Estimate: The project can be developed with near-zero cost using free tiers, with only optional expenses for a domain name.

6 Conclusion

The medical prediction website successfully delivers a platform where users can register, log in, submit their blood report data, and receive predictions for HCV stages using a pretrained machine learning model. Users can view their prediction history, while administrators have access to all prediction records for monitoring purposes. The integration of FastAPI, Next.js, Supabase, and Docker has resulted in a lightweight, efficient, and scalable system.

During development, several lessons were learned and challenges encountered. Implementing the backend API with FastAPI highlighted the importance of asynchronous programming to efficiently handle multiple requests. Integrating the machine learning model required careful preprocessing and validation to ensure accurate predictions. Designing the database schema and optimizing queries in

Supabase was crucial for reliable storage and retrieval of prediction data. Deploying the application with Docker and on cloud platforms presented challenges related to environment configuration and dependency management. Additionally, managing overlapping frontend and backend development tasks emphasized the need for careful project scheduling.

For future improvements, the system could incorporate enhanced user authentication mechanisms such as OAuth or multi-factor authentication to strengthen security. Visualization features could be added to display prediction trends over time for each user, and the machine learning model could be continuously retrained with anonymized new data to improve accuracy. Notifications or email summaries could be implemented to inform users of new predictions, and the platform could be expanded to support additional medical conditions or lab tests. Finally, the deployment process can be further improved with automated CI/CD pipelines and monitoring tools to ensure reliability and maintainability. This project lays a solid foundation for an efficient, scalable, and user-friendly medical prediction system while providing opportunities for future enhancement.