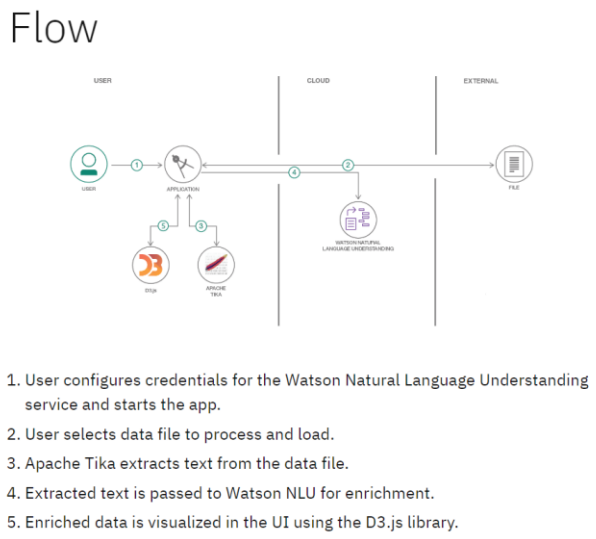
Project Design Phase-II
Data Flow Diagram & User Stories

Date	29 June 2025
Team ID	LTVIP2025TMID46471
Project Name	Hematovision : advanced blood cell classification using transfer learning
Maximum Marks	4 Marks

Data Flow Diagrams:

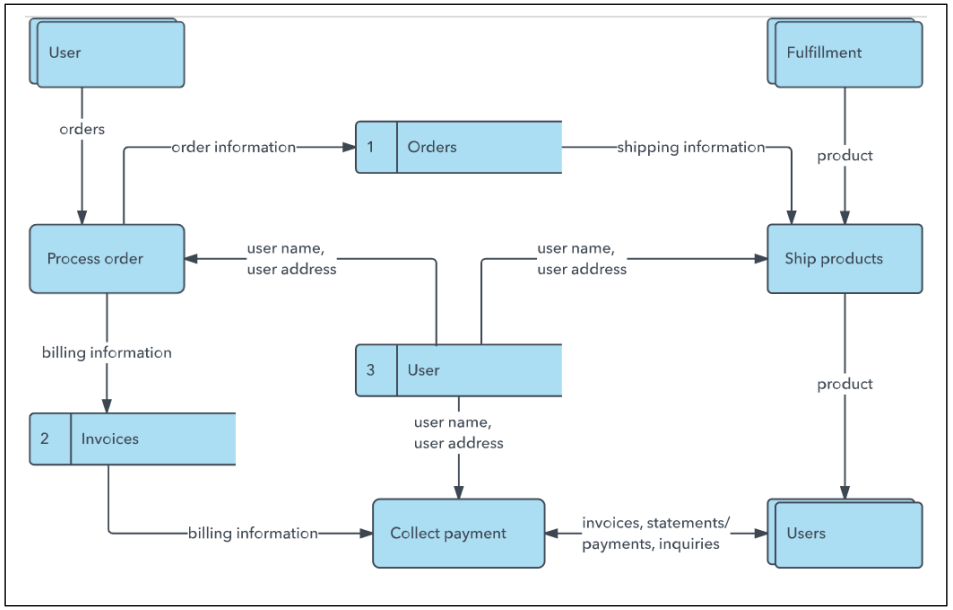
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

Example: (Simplified)



1. User configures credentials for the Watson Natural Language Understanding service and starts the app.
2. User selects data file to process and load.
3. Apache Tika extracts text from the data file.
4. Extracted text is passed to Watson NLU for enrichment.
5. Enriched data is visualized in the UI using the D3.js library.

Example: DFD Level 0 (Industry Standard)



Data Flow Diagram (DFD)

Level 0: System Overview

This Data Flow Diagram (DFD) represents the main entities and their interactions within the blood cell classification system.
Entities:
1. User: Uploads the blood cell image.
2. Web Interface: Displays the image upload page and result.
3. Flask Backend: Handles requests, invokes the trained model, and sends the result.
4. Blood Cell Classifier (MobileNetV2): Processes the image and classifies the blood cell type.
5. Model Storage: Stores the trained model (Blood_Cell.h5).

Data Flows:
1. User uploads image → Web Interface → Flask Backend.
2. Flask Backend sends image → Blood Cell Classifier.
3. Blood Cell Classifier returns prediction → Flask Backend.
4. Flask Backend returns result → Web Interface → User.

Level 1: Flask Backend Process

This diagram zooms into the Flask Backend, detailing its data processing.
Entities:
1. User: Initiates the image upload and receives the classification result.
2. Flask Backend: Receives the image, invokes the model, and returns the result.

Processes:
1. Upload Image: User uploads the image to Flask.
2. Process Image: Flask sends the image to the classifier.
3. Return Result: Flask sends the result back to the Web Interface.

Data Stores:
1. Model Storage: Stores Blood_Cell.h5.

Data Flows:

1. User uploads image → Flask Backend.
2. Flask Backend sends image to the model → Blood Cell Classifier.
3. Blood Cell Classifier sends the classification result → Flask Backend.
4. Flask Backend returns result to Web Interface → User.

User Stories

User Stories

User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
USN-1	As a user, I can upload an image of a blood cell to the application for classification.	The image is uploaded successfully. The system sends the image for classification.	High	Sprint-1
USN-2	As a user, I want to receive a prediction of the blood cell type after uploading an image.	The prediction is displayed correctly. The user can view and interpret the result.	High	Sprint-1
USN-3	As an administrator, I can access the system logs to monitor the prediction system’s performance.	Logs are available and display prediction statistics. The system records errors or exceptions.	Medium	Sprint-2

USN-4	As a user, I can upload another image without needing to refresh the page.	The page clears previous results and allows uploading a new image.	Medium	Sprint-1