Design Structure Matrix (DSM) Development

Adel Ozdem

Department of Computer Science, University of La Verne

Comps

Prof. Juan Rodriguez

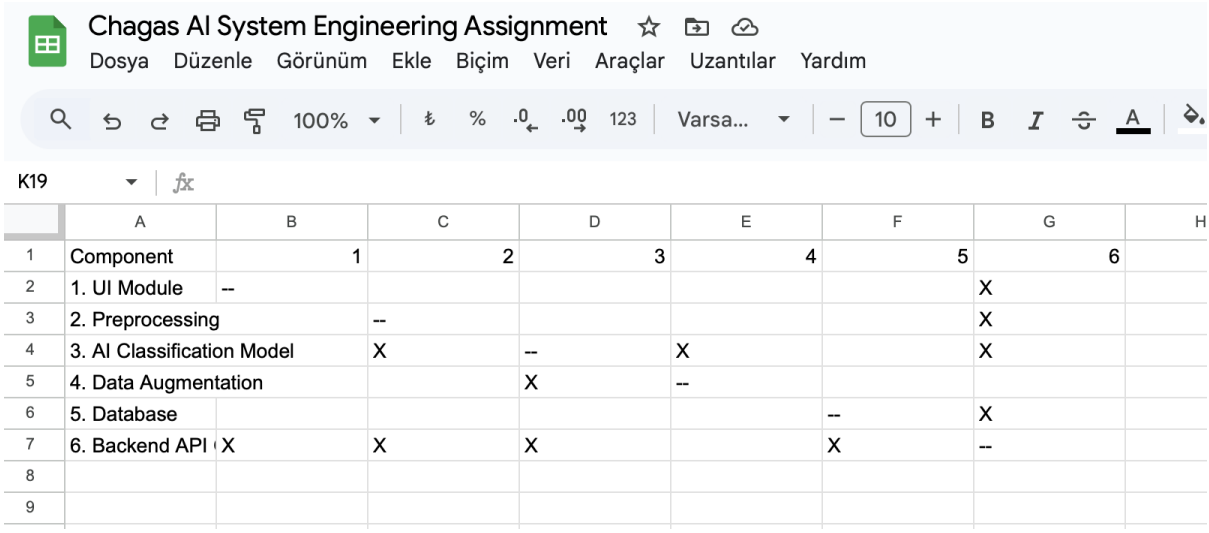
January 13, 2026

Problem 1: Design Structure Matrix (DSM) Development

The following DSM maps the relationships between the key components of the **Chagas AI System**. This system is designed to analyze photos of *Triatoma dimidiata* to detect infection status, regardless of background noise or lighting conditions.

Identified Components:

1. **UI Module (Flask-based Web Interface):** Manages image uploads and displays AI prediction results to the user.
2. **Preprocessing (Background & Light Normalization):** Removes backgrounds and normalizes lighting to ensure the AI focuses solely on the bug features.
3. **AI Classification Model (MobileNetV2):** The core engine trained to classify species and predict infection status (Infected/Uninfected/Unidentified).
4. **Data Augmentation Engine:** Generates variations in light and orientation to compensate for a limited dataset and improve robustness.
5. **Database (SQLite):** Stores bug metadata, coordinates (latitude/longitude), and AI confidence scores.
6. **Backend API Gateway:** Acts as the communication bridge, coordinating requests between the UI, AI model, and storage.



	A	B	C	D	E	F	G	H
1	Component	1	2	3	4	5	6	
2	1. UI Module	--					X	
3	2. Preprocessing		--				X	
4	3. AI Classification Model		X	--	X		X	
5	4. Data Augmentation			X	--			
6	5. Database					--	X	
7	6. Backend API	X	X	X		X	--	
8								
9								

Nature of Dependencies:

- **UI Module → API Gateway:** The UI sends image data to the API to initiate analysis.
- **Preprocessing → AI Model:** The AI model is dependent on the Preprocessing module to receive focused, background-free images for accurate inference.
- **Data Augmentation → AI Model:** The model's training and robustness are directly improved by the variations created by the augmentation engine.
- **Backend API Gateway → Multiple:** The API acts as the bridge for the entire system, coordinating database updates and AI prediction calls.

Problem 2: Reflective Questions

1. What challenges did you encounter while developing the DSM? The primary challenge was defining the functional boundary between the **Preprocessing Module** and the **AI Model**. Determining whether background removal should be a standalone unit or an integrated neural network layer required careful system architecture analysis to ensure the model performs well in both field and lab conditions.

2. What did you learn from mapping the system interactions using the DSM approach? Mapping these interactions revealed that the **Backend API** is the system's critical "Hub." I also learned that the AI model's accuracy is highly dependent on the **Preprocessing stage**; without normalized input, the model's ability to distinguish infection status would fail regardless of the algorithm's complexity.

3. If there were dependencies in the DSM, how would you handle issues if one component were to change or fail? To mitigate component failure, I implemented "Fallback Logic" in the code. If the AI model fails to load, the system triggers a simulation to prevent a total crash. For database failures, the system can cache data locally. Because of the modular DSM design, I can upgrade the AI model with new dataset variations without needing to rewrite the UI or the database schema.