ESOF 322: Project 1 - Cold Case Database

Jacob Coleman, William Jardee, Fletcher Philips, Megan Steinmasel

November 10, 2022

System Description: In the world of criminal justice, it is not a rare occurrence that the truth is thinly veiled by a lack of gathered evidence or proper techniques to analyze current data. A prime example of this is the aid of DNA testing in prosecution. The proposed software system will collect old and new evidence into one database, organize it, and present it to domain specialists through an easy-to-use website. The initial construction of this project is humble but aims to create a skeleton for more sophisticated analysis techniques to be built on. It is assumed that there are two primary ways to interact with the system: through back-end control of the database operations and layman's interpretation of the data in the form of a website; these are the database engineer and website user, respectively. Throughout this whole document, it is assumed that the web servers and hardware maintenance are outside the scope of the software; however, it would make logical sense for the same database engineer to be educated on that system and in charge of it completely as there will likely be an error that occurs between the two systems.

Database Engineer: An individual knowledgeable of how the system works and responsible for maintenance and updates.

Website User: A front-end user that likely lacks data science knowledge traversing the web system.

User Stories

Epic: As a criminal justice professional, I want a general database that can hold, sort, and present data according to a variety of different queries and eventually use state-of-the-art algorithms to help me succeed in my job.

User Story 1 (Megan Steinmasel): As a website user, I want a functional web interface that includes a navigation menu and search bar so that I can access information easily.

User Story 2 (William Jardee): As a website user, I need the processing of new data to find important features and calculate relevant statistics to be done automatically, so it is friendly to someone that is not a data scientist.

User Story 3 (Jacob Coleman): As a website user, I want a data visualization technique that is intuitive and accurately shows patterns in the data.

User Story 4 (Fletcher Philips): As a database engineer, I want to have a database so that I can store cold cases inside of it.

User Story 5 (Fletcher Philips): As a database engineer, I want the data to be manipulable so that the admin can insert and delete data from the database.

1 UseCase Diagram

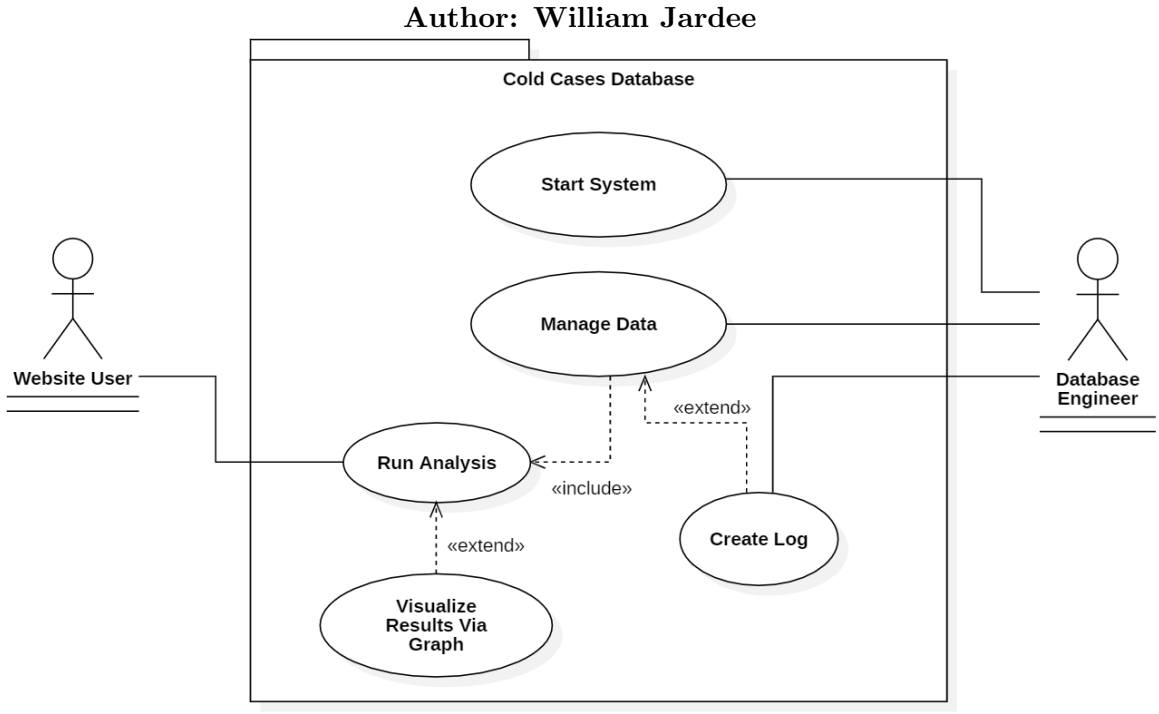


Figure 1: UseCase Diagram for the Cold-Cases Database system.

1.1 Textual Descriptions

Author: Fletcher Philips

Name:	Create Database/Start System
Description:	A database needs to be created and connected to the web framework we choose.
Related Requirements:	User Story 4
Preconditions:	Basic web infrastructure has been created. A small sample set of cold cases is ready to be inserted.
Successful end condition:	Basic database has been initialized with a substructure. Data has been inserted into the database.
Failed end condition:	Data cannot be inserted into the database.
Actors:	Database Engineer
Basic Flow of Events:	1. Create Database. 2. Connect a database to the web framework. 3. Insert Initial Data into Database.
Extensions/Exceptional Flow of Events	1. Database fails to build properly. 2. The engineer is notified that an error has appeared. 3. Write to log file. 4. Engineer is ejected from the system.

Author: William Jardee

Name:	Manage Data
Description:	Data needs to be inserted, deleted, and manipulated. Related to this, there must be an appropriate Graphical User Interface. The data will connect directly with the database.
Related Requirements:	User Story 2, User Story 5
Preconditions:	The user is logged on and has gained access rights according to their credentials.
Successful end condition:	Data is successfully manipulated, and a success message is received from the source.
Failed end condition:	Requested task is outside of credentials. Success message not received. Invalid new data.
Actors:	Database Engineer
Basic Flow of Events:	1. Actor selects the action they wish to commit and what to commit it on. 2. Action is tested against credentials. 3. Success/Fail state is determined. 4. Flow is complete and prompts the user for the next action.
Extensions/Exceptional Flow of Events	1. Conflict happens. 2. Reject any attempted changes and revert to the last viable state. 3. Notify the actor that there has been an error and write to the log file. 4. Flow is complete and prompts the user for the next action.

Author: William Jardee

Name:	Run Analysis
Description:	Graphic User Interface for viewing data and selecting data visualization.
Related Requirements:	User Story 2, User Story 3
Preconditions:	The website user has signed into the website and accessed the drop-down menu to access the page.
Successful end condition:	A valid group of data is selected, a visualization format is selected, and the servers are available for the request.
Failed end condition:	One of the above three requirements has been violated, and the window is closed.
Actors:	Website User
Basic Flow of Events:	1. A query selection screen is presented, and the desired data is collected. 2. The type of visualization is selected from a dropdown menu. 3. The request is sent to the "Visualize Results via Graph" action.
Extensions/Exceptional Flow of Events	1. One of the system-side error states was reached. 2. Report the error to the actor. 3. Write error to log file. 4. Return actor back to the homepage.

Author: Jacob Coleman

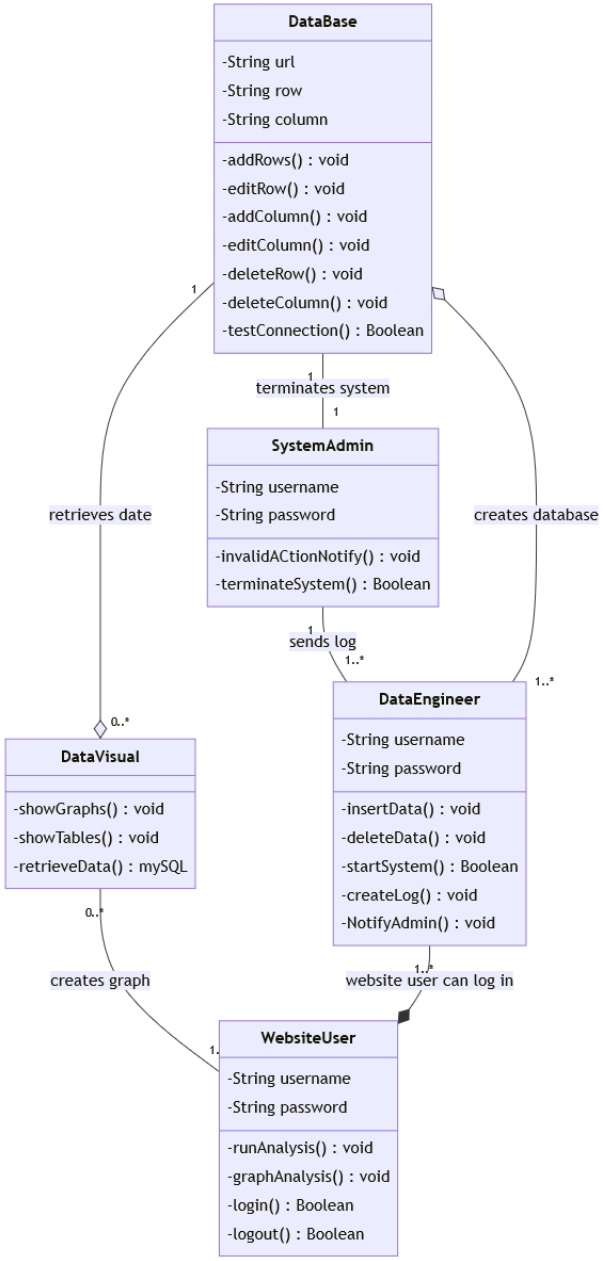
Name:	Visualize Results via Graph
Description:	Displays selected graphs from the gathered data
Related Requirements:	User Story 3
Preconditions:	The website user has signed into the website and accessed the drop-down menu to access the page. Some data has been selected for analysis, and the desired type of graph has been selected.
Successful end condition:	The graphs are displayed on a separate page to view.
Failed end condition:	Fails to display any graphs.
Actors:	Website User
Basic Flow of Events:	1. The website user goes through the drop-down menu process. 2. The website user chooses to view graphs via the drop-down menu. 3. The system will display any applicable graphs.
Extensions/Exceptional Flow of Events	1. The graphs fail to display. 2. The User is notified that the user cannot access graphs via a notification. 3. Write to log file. 4. User is sent back to the main navigation system.

Author: William Jardee

Name:	Create Log
Description:	A log file should be kept to track flow and diagnose errors and suspicious behavior.
Related Requirements:	Catch all location for all errors (no specific user story)
Preconditions:	The system has been started effectively, and there is a safe place to store a text file (log file).
Successful end condition:	Data can be saved to the log file.
Failed end condition:	Data cannot be safely saved to a log file.
Actors:	Database Engineer
Basic Flow of Events:	1. Write to file recent activity. 2. Flag any invalid actions prompt "write to log file."
Extensions/Exceptional Flow of Events	1. Notify the system admin of the issue and include error information. 2. Terminate all systems until the issue is resolved.

Class Diagram

Author: Fletcher Philips (Mermaid: William Jardee)

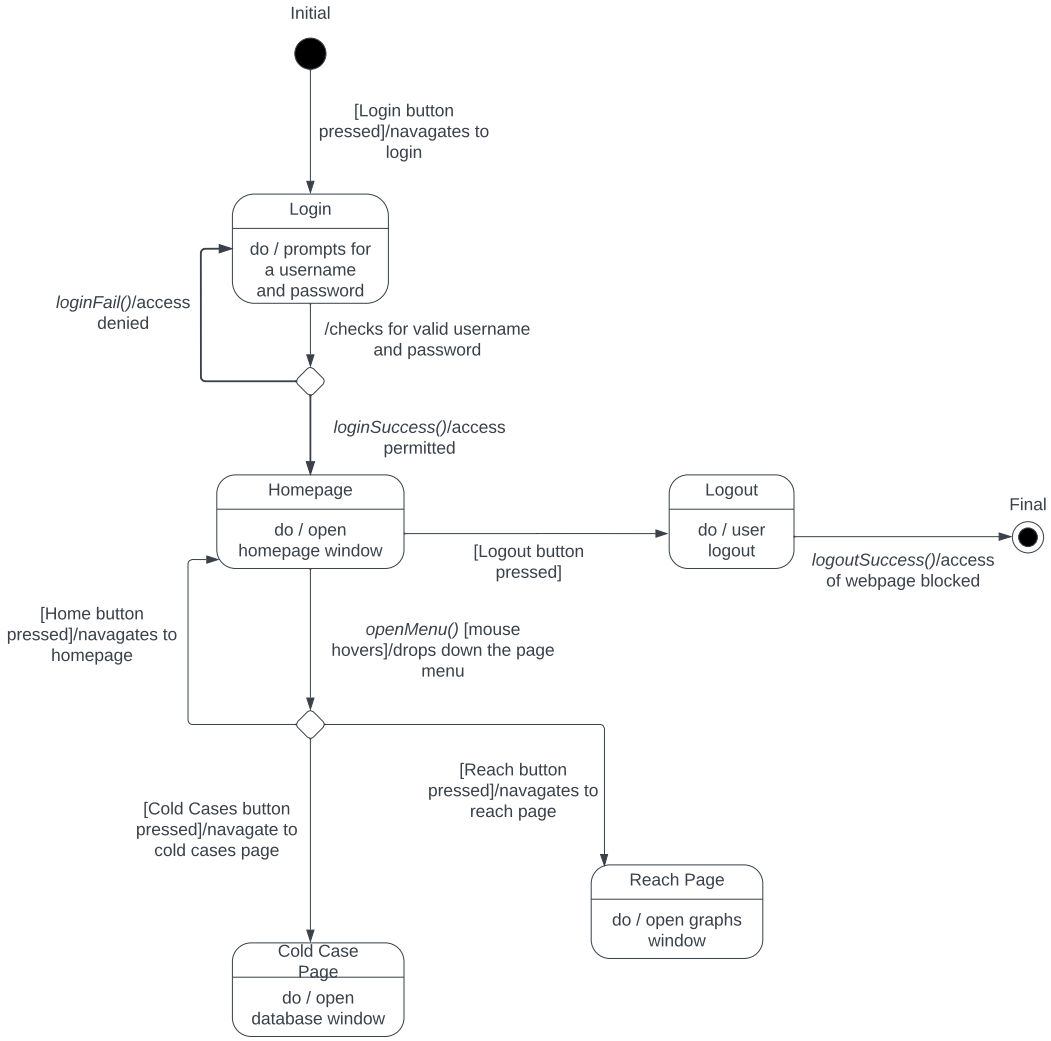


{If this image is difficult to read, refer to the origin: mermaid.live image, or the editing link here mermaid.live editor.}

Figure 2: Class Diagram for the Cold-Cases Database system.

State Chart Diagram

Author: Megan Steinmasel (Mermaid: William Jardee)

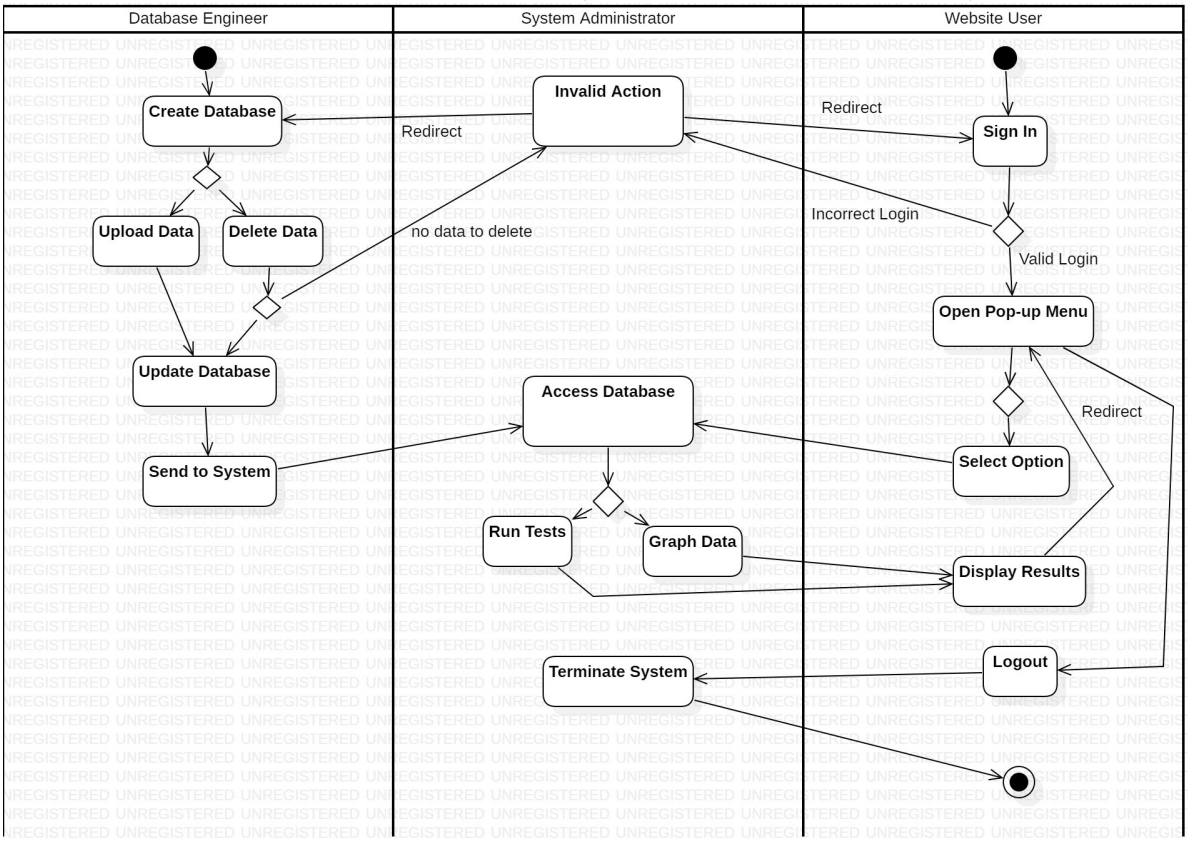


{If this image is difficult to read, refer to an online rendition: mermaid.live image, or the editing link here mermaid.live editor.}

Figure 3: State Chart Diagram for the Cold-Cases Database system.

Activity Diagram

Author: Jacob Coleman (Mermaid: William Jardee)



{If this image is difficult to read, refer to an online rendition: mermaid.live image, or the editing link here mermaid.live editor. Notice that the Mermaid version has numerous formatting issues. Mermaid does not support Activity Diagrams yet.}

Figure 4: Activity Diagram for the Cold-Cases Database system.