AnthroFace: Requirements, Design, and Implementation

Skylar Glock, Kealia Perrine, Henry Sturm, Gabrielle Talavera, Mari Trombley, Cortney Hulse, Kyra E. Stull, and Frederick C. Harris, Jr.

University of Nevada, Reno Reno, NV, USA, 89557
Fred.Harris@cse.unr.edu

Abstract

AnthroFace is a rib fracture data collection and analysis web application. Currently researchers at the University of Nevada, Reno (UNR) are manually collecting data using `.csv` files and do not have it in a centralized location. The purpose of this software is to provide researchers an efficient way to collect and record relevant data and make meaningful interpretations from the dataset. The main features of AnthroFace can broadly be placed into three categories, a quality user interface system to efficiently collect and record relevant data, a database to store collected information, and tools to help the user make meaningful interpretations from the dataset.

1 Introduction

AnthroFace is a web application that allows users to efficiently enter new rib fracture data through an intuitive and convenient user interface system. This application uses modern web technologies to assist researchers at the University of Nevada, Reno (UNR) in data collection and analysis. AnthroFace has been in development by its authors since 2020 and will be continuing to be developed further to implement additional features.

AnthroFace allows users to manually add all rib fracture data along with relevant demographic data or upload properly formatted data that is stored in a .csv file for easy mass entry of already collected rib fracture data. These functionalities in AnthroFace remove the need to upload rib fracture data into Microsoft Excel sheets. AnthroFace stores the data uploaded by users in an online database that will securely hold all of the rib fracture data to be accessed by other users. Multiple users can share their full dataset easily and have a centralized full database from which to pull data from for analysis. The database is filterable for users to find either specific cases or cases that are most similar to a user designated case in rib fracture type or through demographics. Lastly, AnthroFace provides users with the ability to easily generate heat maps or generate charts that show meaningful statistical analysis so that users can better visualize the data that is collected and discover meaningful trends in data.

Before the creation of AnthroFace, researchers in forensic anthropology at UNR would manually enter all patient information through Excel files which is very inefficient, tedious, error prone, and does not allow for all information to be placed in one centralized location. AnthroFace was created to allow for an efficient way to collect and analyze the data which will help researchers understand rib fracture patterns and the variables that influence their occurrence

and severity. Upon introducing AnthroFace to the researchers, data collection and analysis has become much more efficient and easier which creates a better experience for researchers at the university.

The rest of this paper is structured as follows: Section 2 describes the motivation for creating AnthroFace and its design. Section 3 explains the application and its features, functionality, and technologies used to create AnthroFace. Section 4 goes over our Conclusion and Future Work.

2 Motivation and Design

The technical specifications for AnthroFace were developed through a series of interviews with our collaborators in anthropology. The AnthroFace development team typically met weekly to discuss the requirements, design, implementation, integration, and operation of the application.The team would also hold check-in meetings with anthropology collaborators to ensure a;; their wants and needs were addressed in the application, as the development team based needs off of their Excel dataset files [4]. The development team itself would also meet to work together weekly to keep the team on track and help with any blockages. These meetings were a vital part to successfully develop AnthroFace. Software development methodologies such as scrum boards and stand-up meetings were used to keep the team on track. The major goals of AnthroFace were to allow users to record all relevant rib data, create a database to house all this data, provide a tool for statistical analysis of the data, and to create graphs and heat maps to visualize the data. These goals are documented through the software specifications listed in Subsections 2.1, 2.2, 2.3 and 2.4.

2.1 Functional Requirements

The following requirements in this subsection are "statements of services the system should provide, how the system should react to particular inputs, and how the system should behave in particular situations" [7]. These functional requirements outline what AnthroFace is capable of doing.

1. AnthroFace shall allow the user to manually upload rib data.
2. AnthroFace shall allow the user to upload a .csv file containing rib data.
3. AnthroFace shall keep a database in a web server.
4. AnthroFace shall hold patient information in a dataset.
5. AnthroFace shall allow the user to add a new patient to the current dataset.
6. AnthroFace shall allow the user to download the current dataset onto their local machine.
7. AnthroFace shall allow the user to filter similar patient cases based on demographic and rib fracture data.
8. AnthroFace shall allow the user to delete a patient case from the dataset.
9. AnthroFace shall create a heat map from all patients in the current dataset to indicate common breaks.
10. AnthroFace shall allow a user to view various heat maps that update when filtered by demographic and rib fracture data.
11. AnthroFace shall allow the user to log in using their credentials.
12. AnthroFace shall allow the user to log out.
13. AnthroFace shall allow the user to modify an existing patient case.

14. AnthroFace shall allow the user to work on multiple different datasets.
15. AnthroFace shall allow the user to delete the dataset that is currently selected.

2.2 Non-Functional Requirements

The following requirements in this subsection are "constraints on the services or functions offered by the system. They include timing constraints, constraints on the development process, and constraints imposed by standards" [7]. These non-functional requirements outline characteristics of AnthroFace.

1. AnthroFace will run on all browsers including Google Chrome, Safari, and Firefox.
2. AnthroFace will have an easily understandable user interface.
3. AnthroFace will have a short learning curve to learning to use the app.
4. AnthroFace will have a sleek and professional user interface.
5. AnthroFace shall have a front-end implemented in React.
6. AnthroFace shall have a back-end implemented in Django.
7. AnthroFace will store database objects based on their case number.
8. AnthroFace shall have heat maps that are encoded and have color gradient.

2.3 Detailed Use Cases

1. **UC-01 DeleteDataset**
   Removes a dataset from the server so that the user is no longer able to access it.
2. **UC-02 CreateNewPatient**
   Switches the application from the main menu to a new screen where data entry can be done for a new patient.
3. **UC-03 FilterExistingPatients**
   Allows the user to filter existing patients by identifying factors.
4. **UC-04 DeleteExistingPatient**
   Removes the selected Patient and all associated data on the local copy of the database.
5. **UC-05 UpdatePatientData**
   This page will prompt the user to input demographic and injury data for the new patient or allow data to be updated on an existing patient.
6. **UC-06 UserLogin**
   On application startup or after logout the user will be prompted to login using a username and password before allowing any other user input.
7. **UC-07 SwitchDataset**
   Allows the user to switch from the dataset currently being used to a new selected dataset.
8. **UC-08 ImportLocalData**
   Allows the user to upload a correctly formatted .csv file to be entered into the currently selected dataset.
9. **UC-09 ExportDatasetLocally**
   Allows the user to download the current dataset as a .csv file.
10. **UC-10 DisplayHeatMap**
    Displays a heat map of common fracture points for the selected dataset and updates this heat map if the dataset is filtered.
11. **UC-11 UserLogout**
    Allows the user to log out of the application prompting the username and password to be re-input before any other input is performed.

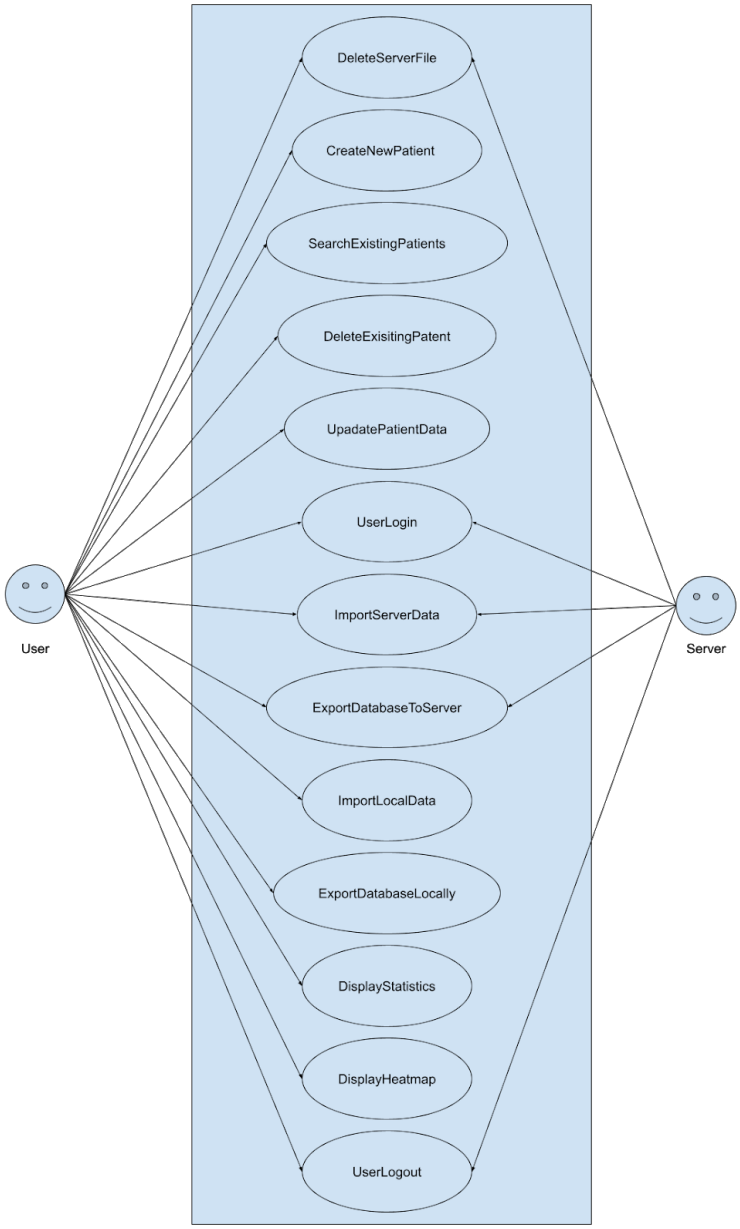

Figure 1: AnthroFace Use Case Diagram

2.4 Traceability Matrix

Traceability policies "define the relationships between each requirement and between the requirements and the system design that should be recorded." [7] The Traceability Matrix, as seen in Figure 2, is the tool used to track the relationship between the requirements and the Use Cases. Organizing these connections is essential in analyzing proposed changes and the impact they have on other parts of the system.

	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	UC10	UC11
R1					x						
R2								x			
R3	x										
R4	x										
R5		x			x						
R6									x		
R7			x								
R8				x	x						
R9										x	
R10			x							x	
R11						x					
R12											x
R13					x						
R14							x				
R15	x										

Figure 2: AnthroFace Traceability Matrix showing relationship between Use Cases (UC) and Requirements (R)

3 Implementation

Upon startup, the AnthroFace application greets users with a login screen where a current user can either log in with their credentials or a new user can create an account. Once a user is logged in they are sent to the home page which allows them to reach any of the other part of the application. The `select dataset` page allows the user to choose between any previous datasets that they have created, or make a new dataset. On this page users are also able to delete their currently selected dataset. On the `database` page users can see at all of the patient information for each patient in their current dataset in table form as shown in Figure 3. Users can also edit and delete specific patients on this page as needed. The `database` page can additionally apply filters as desired to only show a subset of patients matching filter requirements. To add new patients the user can either go to the `new patient` page and add users in one at a time or if there is an existing spreadsheet with patient data it can be mass uploaded on the `import data` page. If the user ever needs to export a dataset back to a spreadsheet the `export data` page

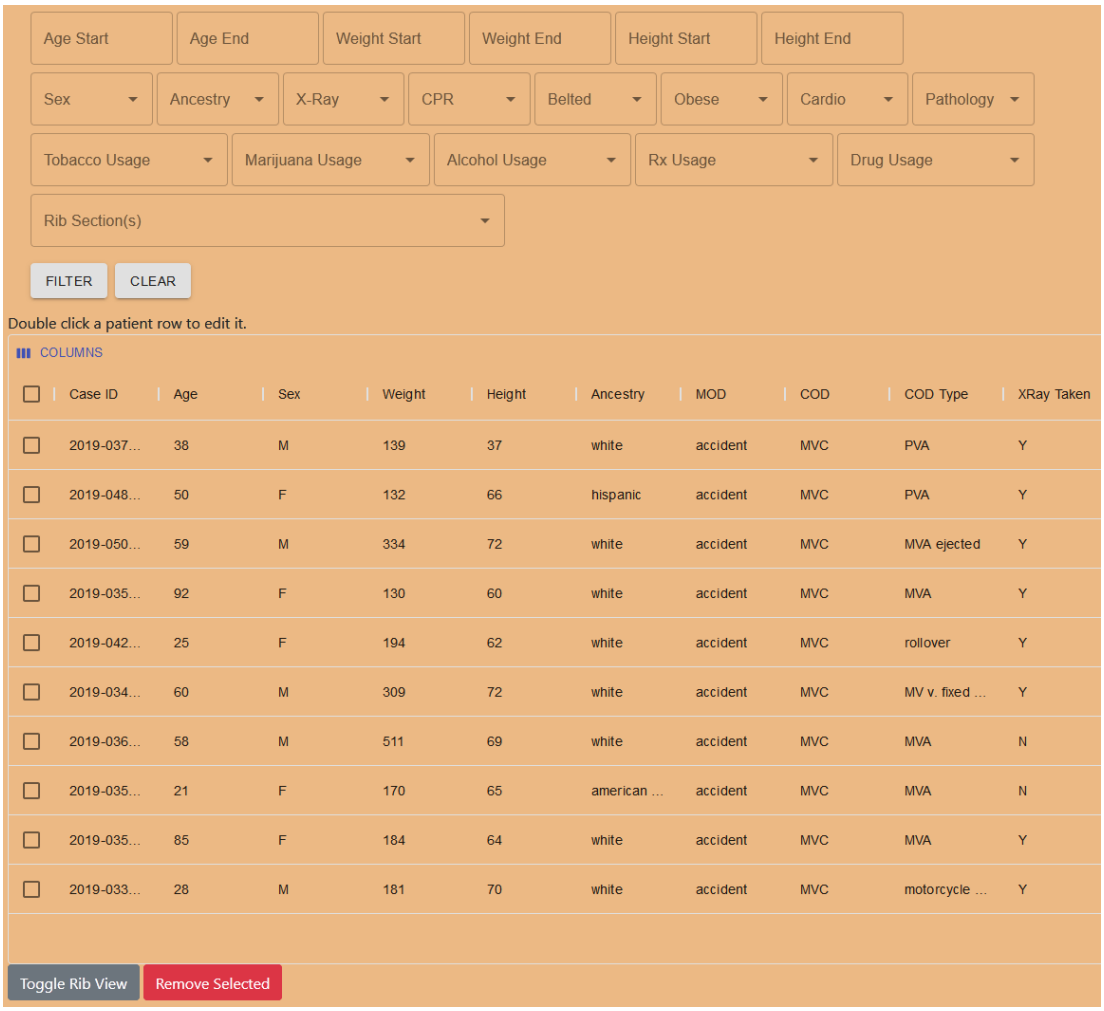

Figure 3: Database

allows the user to download a spreadsheet of the dataset back to their local machine. Finally there is a `heatmap` page where the user can view either a rib heat grid [3] or rib heat image [6] as seen in Figures 4 and 5. As with the `database` page, the `heatmap` page can have filters set to change the patients included in the rib heat grid or rib heat image.

3.1 Architectural Design

Figure 6 is an activity map presenting an overview of the web application's user actions. The course of action taken and what new higher level access is given to the user can be determined and what course of action is needed to access the data. Figure 7 is a sitemap presenting an overview of the options presented to the users on the web application and what page presents varying options for users. Figure 8 pulls this all together in the context diagram for AnthroFace.

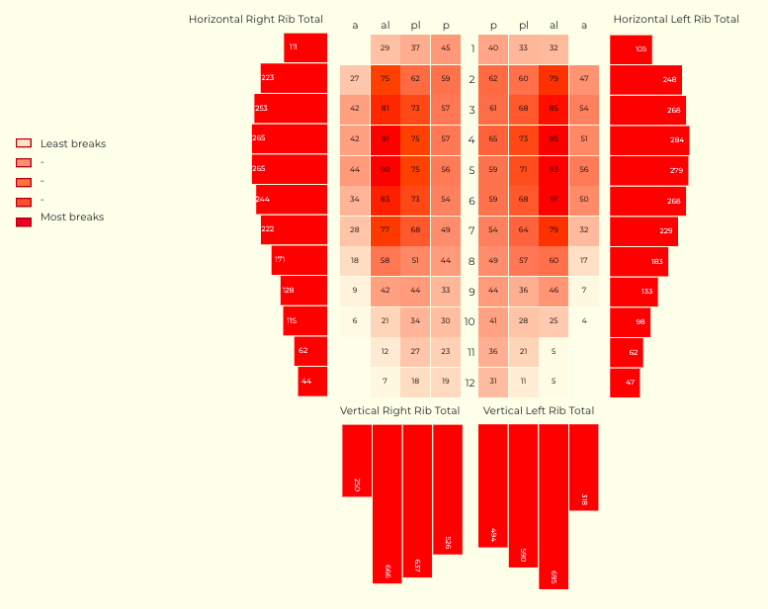

Figure 4: Rib Heat Grid

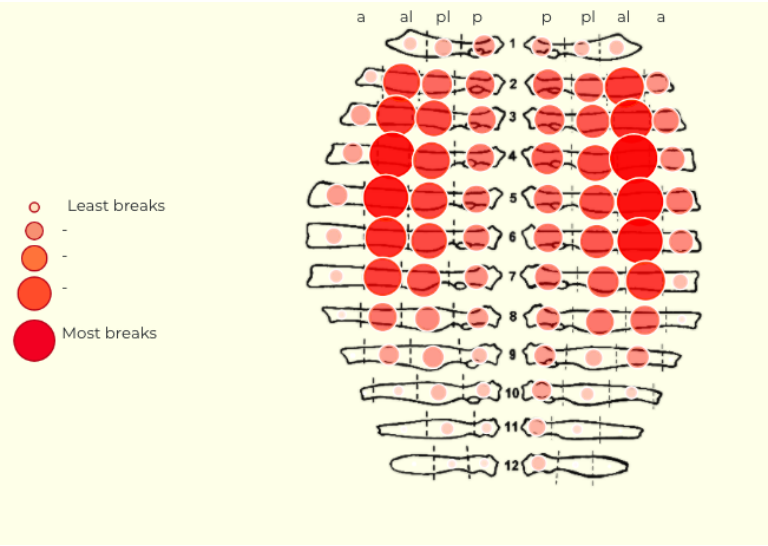

Figure 5: Rib Heat Image

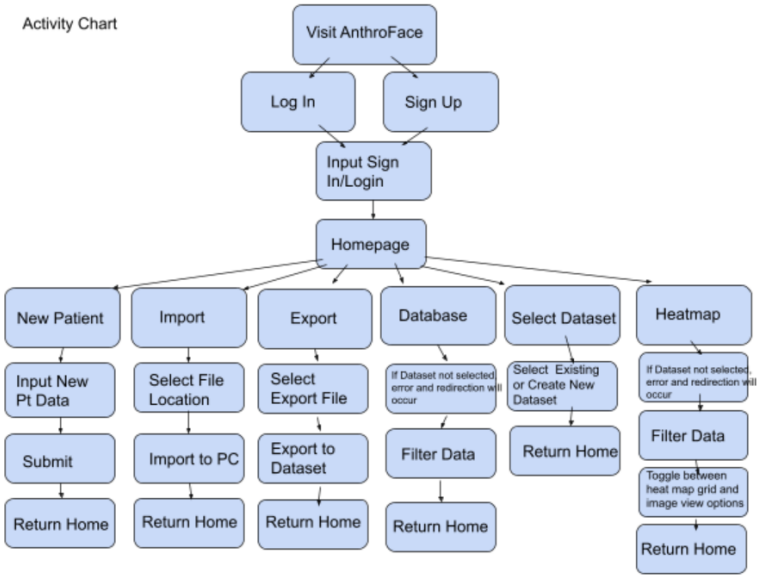

Figure 6: Activity Chart

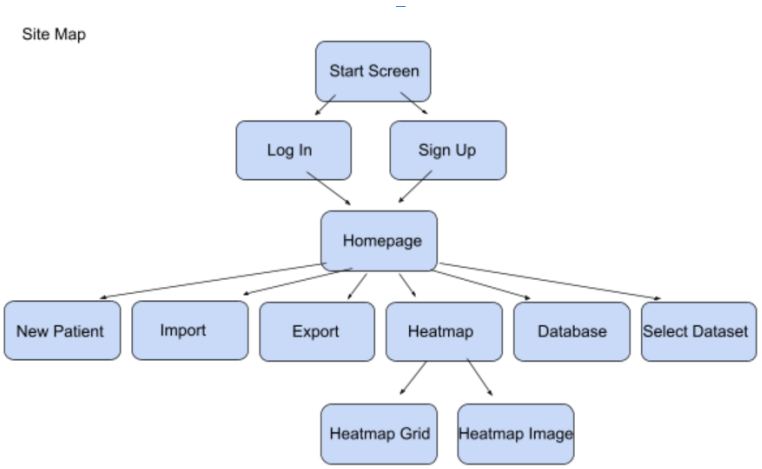

Figure 7: Sitemap

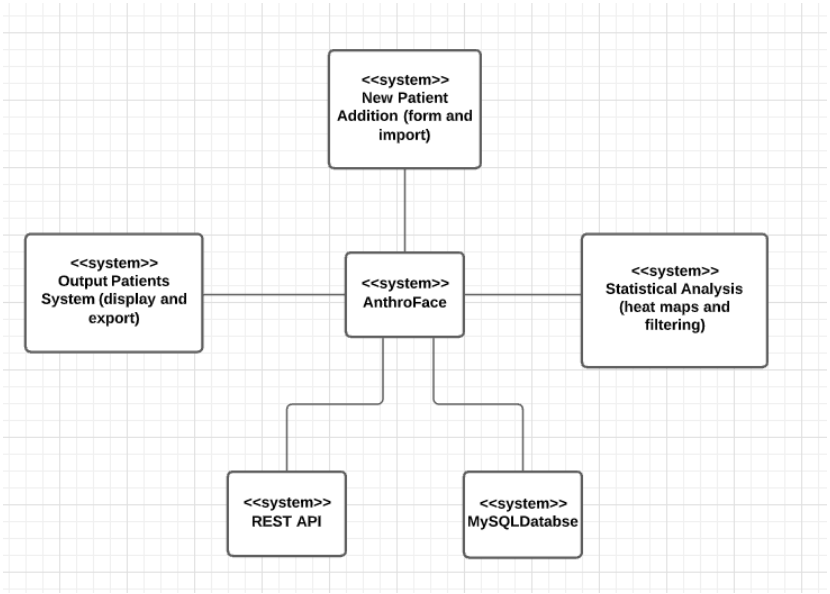

Figure 8: AnthroFace Context Model

3.2 Technologies Used

Visual Studio: Visual Studio is one of the technologies used to create AnthroFace. Visual Studio is an IDE that provides an interpreter and a great source code editor. The development team used Visual Studio to develop the code with a Django backend and React frontend using Python, JavaScript, HTML, and CSS. There is also auto-complete and debugging support for integrated with Visual Studio. Visual Studio had many extensions that made it easy and efficient to use including a live share where programmers can work on the code at the same time [5].

React: The frontend of AnthroFace was created using React because of its simplicity and ability to efficiently create interactive user interfaces. The declarative views used by React allow for fast updating and re-rendering of components as data changes, which is necessary for functionalities such as applying filters to the rib heat grid [2].

Django: The backend of AnthroFace was created using Django because of its easy to set up framework and vast array of built in tools to help with web development. The Django REST framework was used to create the API for transferring data between the backend and frontend of AnthroFace. Django's user authentication system was also used to setup logins for Anthroface [1].

SQLite: SQLite was used to hold the various data tables used by AnthroFace including user and patient tables. Interactions with SQLite including data read and write operations are all facilitated through Django's models system [8].

4 Conclusions and Future Work

AnthroFace allows anthropology researchers to efficiently collect and analyze rib fracture data. The heat maps allow for visual analysis of clusters and patterns which can be filtered by a number of demographics. The application uses a React frontend in combination with a Django backend and SQLite for the database.

AnthroFace will eventually have an added statistical analysis feature. This will allow users to upload pictures of rib fractures which will be compared to other cases in the database through cosign similarity. The application will also have an added "recommender" system which will find patterns in the dataset in order to suggest similar cases.

AnthroFace could be modified to be used for many different works, particularly with other bone breaks or fractures besides ribs. Just as well, while the creation of the application was targeted for anthropology researchers, AnthroFace could have possible uses in the healthcare industry.

Acknowledgments

This material is based in part upon work supported by the National Science Foundation under grant number IIA-1301726. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Django Software Foundation. Django: The web framework for perfectionists with deadlines. https://www.djangoproject.com/, 2021. (last accessed May 2021).

[2] Facebook Open Source. React: A javascript library for building user interfaces. https://reactjs.org/. (last accessed May 2021).

[3] Arun Ghosh. react-heatmap-grid. https://www.npmjs.com/package/react-heatmap-grid, 2020. (last accessed May 2021).

[4] Cortney Hulse, Kyra Stull, and Ashley Weaver. Rib fracture frequency and location using vehicular crash data. *Forensic Anthropology*, 2(1):1–8, 2019.

[5] Microsoft. Visual studio. https://visualstudio.microsoft.com. (last accessed May 2021).

[6] Benoit Sida. react-image-mapper. https://www.npmjs.com/package/react-image-mapper, 2017. (last accessed May 2021).

[7] Ian Sommerville. *Software Engineering*. Pearson, 10th edition, 2015.

[8] SQLite Consortium. Sqlite. https://www.sqlite.org/. (last accessed May 2021).