Software Architecture Description for

Afet Bilgi

Emre Geçit, Baran Yancı

May 31, 2023

Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose and Objectives

The purpose of afetbilgi.com is to provide a centralized platform for users to access information about natural disasters and emergency response procedures. The website aims to be a reliable source of information that can help users prepare for, respond to, and recover from disasters.

1.2 Scope

The scope of afetbilgi.com includes the following:

- Provide up-to-date information about natural disasters, including their causes, impacts, and possible response procedures.
- Provide important phone numbers, or the locations of important places or services to the people affected by the earthquake.
- Provide information about donation centers; such as blood, stem cells, money and so on to the people who want to help the people affected by the earthquake.
- Offer people to create help points and share the locations of the help points, with the help of the system.

The scope of afetbilgi.com does not include the following:

- Providing direct and instant communication services during a disaster event (this is the responsibility of official emergency response organizations).
- Offering financial assistance or physical aid during a disaster event (this is the responsibility of the official relief organizations).

1.3 Stakeholders and Concerns

The stakeholders of afetbilgi.com include the following:

- **Users:** They are concerned with accessing accurate information about natural disasters, preparing for emergencies, and communicating with others during a disaster event.
- **Emergency responders:** They are concerned with having access to timely and accurate information about disaster events to respond effectively.
- **Information providers:** People who voluntarily provide information about the earthquake. They can reach out to the maintainers of the system to provide information.
- **Maintainers / Developers:** People who maintain and develop the system. They develop the software and maintain it by verifying new information and adding it to the system.

2 References

The following references were consulted during the development of this SAD:

- ISO/IEC/IEEE 42010:2011, Systems and software engineering – Architecture description
- Rozanski, N., & Woods, E. (2011). Software systems architecture: working with stakeholders using viewpoints and perspectives. Addison-Wesley Professional.

3 Glossary

The following terms are used throughout this SAD and have the following meanings:

- **Disaster:** A sudden, catastrophic event that causes widespread damage or loss of life. Examples include earthquakes, hurricanes, floods, and wildfires.
- **Emergency responder:** A person who is trained to respond to emergencies and provide assistance to those in need. Examples include police officers, firefighters, and paramedics.
- **ISO/IEC/IEEE 42010:** A standard for software architecture description that provides a framework for describing the architecture of a software-intensive system.
- **Relief organization:** An organization that provides aid and support to communities affected by disasters. Examples include the Red Cross and UNICEF.
- **Scope:** The boundaries of a project or system, including what is included and what is excluded from consideration.
- **Stakeholder:** A person or group who has an interest in the success of a project or system. Examples include users, customers, and investors.
- **Viewpoint:** A perspective on the architecture of a system that emphasizes certain aspects or concerns. Examples include functional, data, and deployment viewpoints.
- **AWS:** Amazon Web Services, a cloud computing platform that provides on-demand access to computing resources.

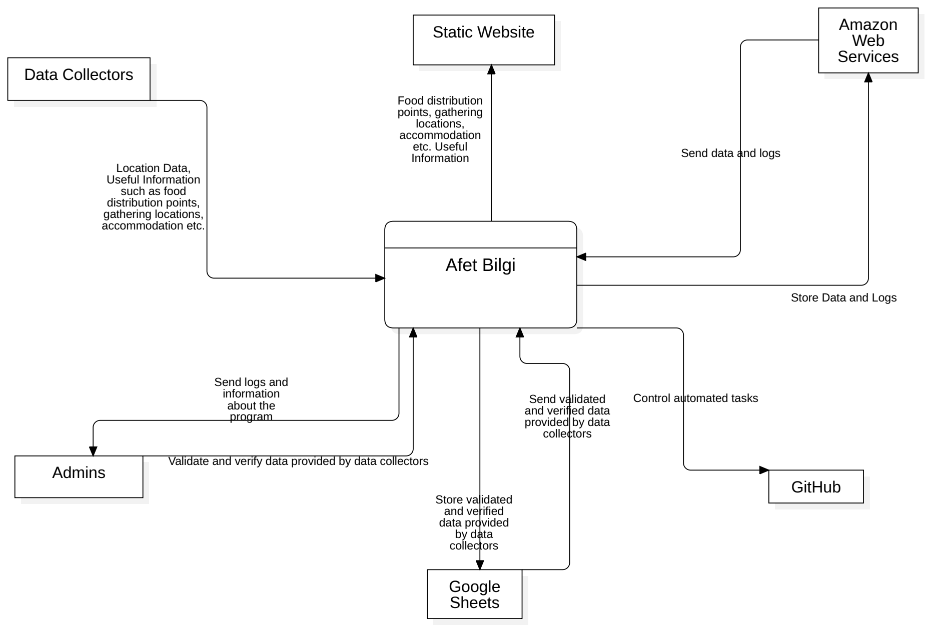
Static Website
Amazon Web Services
Data Collectors
Food distribution points, gathering locations, accommodation etc. Useful Information
Send data and logs
Location Data, Useful Information such as food distribution points, gathering locations, accommodation etc.
Afet Bilgi
Store Data and Logs
Send logs and information about the program
Send validated and verified data provided by data collectors
Control automated tasks
Admins
Validate and verify data provided by data collectors
GitHub
Store validated and verified data provided by data collectors
Google Sheets

Figure 1: Context Diagram

4 Architectural Views

4.1 Context View

4.1.1 Stakeholders' Uses

There are four main stakeholders of the system: the users, emergency responders, information providers and the programmers. The users use this view to understand the general structure of their interaction with the system. The emergency responders use it to understand the general structure of the system to plan their response to the disaster. Information providers use it to supervise the processes their data goes through. Programmers use it to understand the general structure of the system to develop and maintain it.

4.1.2 Context Diagram

As shown in the diagram, users can access the website through a web browser or mobile device. They can view information about natural disasters, emergency response procedures, and other related content. Administrators have

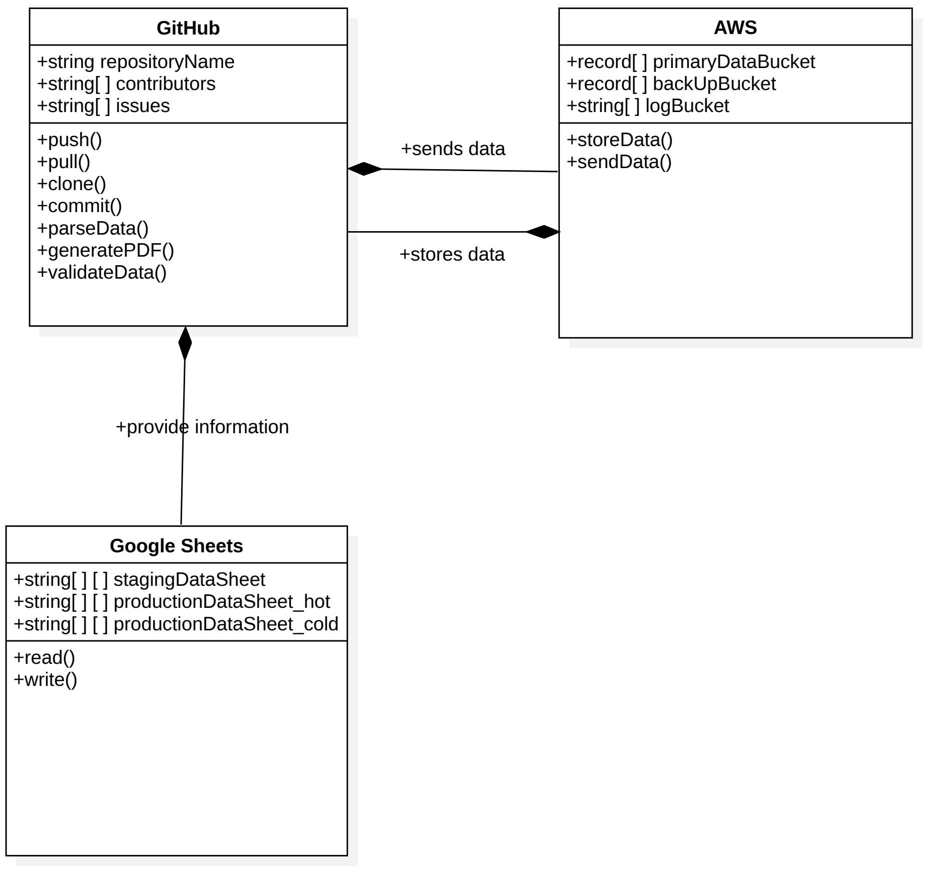

Figure 2: External Interfaces

additional privileges and can manage user accounts, update content on the website, and monitor site analytics.

Third-party services are also used to provide additional functionality on the website. For example, a map API is used to display maps of affected areas and help users locate nearby emergency resources.

Overall, the Context Diagram provides a high-level overview of how afetbilgi.com interacts with its external environment and stakeholders. This information is useful for understanding the scope of the system and identifying potential risks or dependencies that may impact its operation.

4.1.3 External Interfaces

As it can be observed from the figure above, afetbilgi.com has multiple external interfaces. GitHub, Information Provider, Information Validator, Developer, Google Sheets, and AWS. The operations given in the diagram can be summarized as follows:

Operation	**Description**
registerInformation()	Data providers can register new pieces of information to the Google Sheets-hosted database.
push()	Collaborators can change the code hosted in GitHub servers.
pull()	Collaborators can fetch the latest version of the source code hosted on GitHub servers.
clone()	Collaborators can download and work on the code.
commit()	Collaborators can submit their changes on the code.
parseData()	GitHub server parses the data stored on AWS.
generatePDF()	Automated scripts hosted on GitHub servers generate PDF files based on the latest information.
validateData()	Automated scripts hosted on GitHub servers validate the data stored on AWS.
storeData()	Automated scripts hosted on GitHub servers store the data on AWS.
sendData()	Automated scripts hosted on GitHub servers request data from AWS and AWS sends the data.
read()	Users can read the information stored on the Google Sheets.
write()	Users can write the information stored on the Google Sheets.

Table 1: Operations on External Interfaces

4.1.4 Interaction Scenarios

4.2 Functional View

4.2.1 Stakeholders' Uses

There are four main stakeholders of the system: the users, emergency responders, information providers and the programmers. The users use this view to understand the capabilities and limitations of the system. The emergency

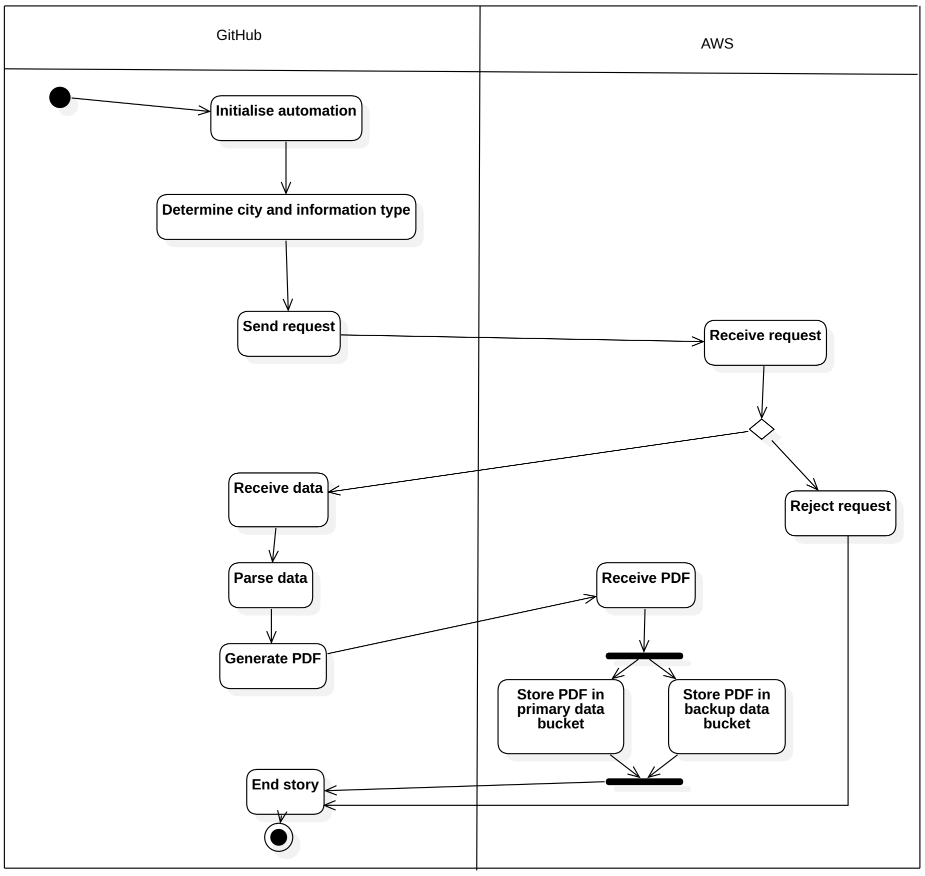

Figure 3: Activity Diagram for GitHub - AWS Interactions

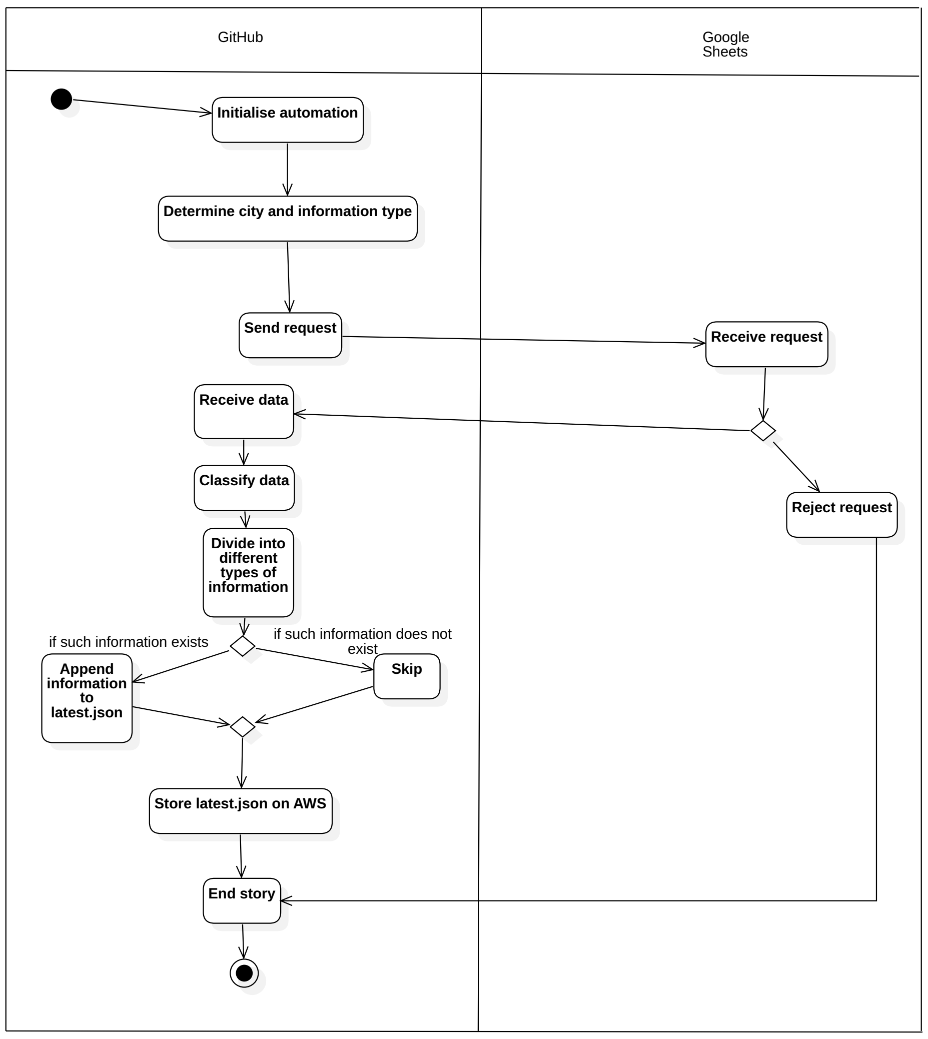
GitHub
Google Sheets
Initialise automation
Determine city and information type
Send request
Receive request
Receive data
Reject request
Classify data
Divide into different types of information
if such information exists
if such information does not exist
Append information to latest.json
Skip
Store latest.json on AWS
End story

Figure 4: Activity Diagram for GitHub - Google Sheets Interactions

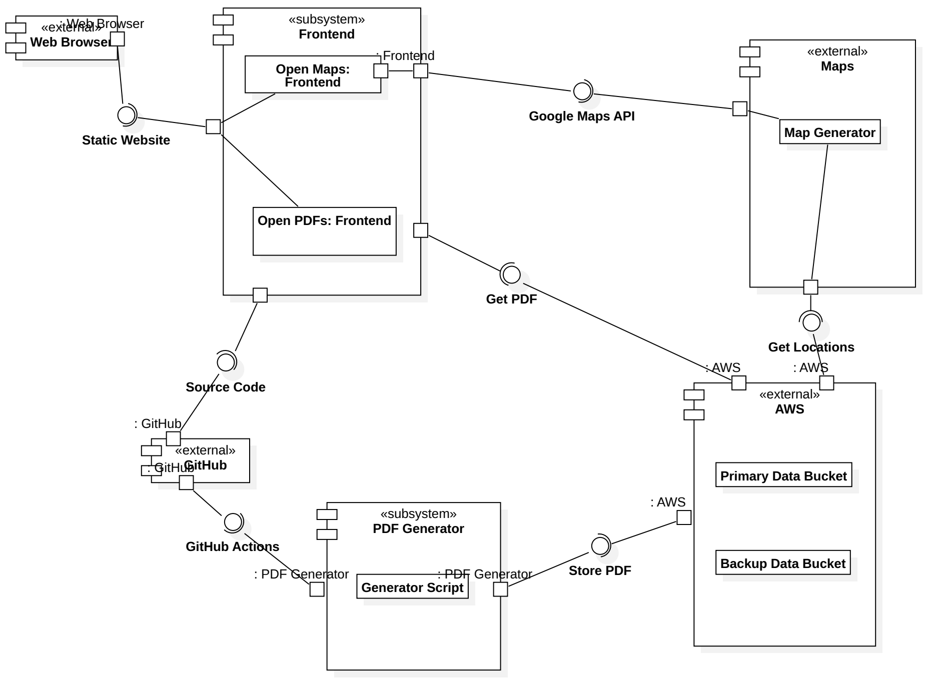

Figure 5: Component Diagram

responders use it to determine how the system can be used to respond to the disaster. Information providers use it to understand what procedures are used and how they operate on their data. Programmers use it to understand the capabilities and limitations of the system to develop and maintain it.

4.2.2 Component Diagram

Our system consists of two main subsystems: the Frontend and the PDF Generator.

- GitHub is an external component that hosts the source code of the system. It is used by the developers to develop and maintain the system. It is also responsible for generating PDF files based on the latest information. For this purpose, it uses the PDF Generator subsystem via GitHub Actions.
- The frontend is the main subsystem of the system. It is responsible for providing the main functionality of the system to the users. It is also responsible for communicating with the AWS-hosted database.

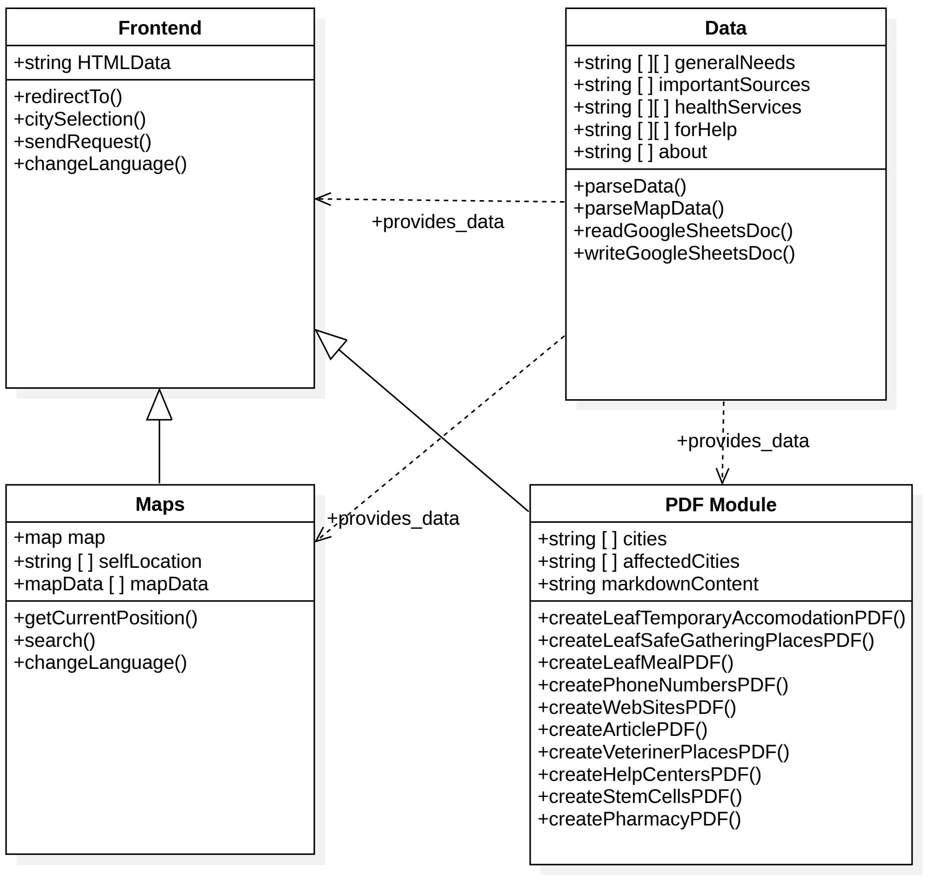

Figure 6: Internal Interfaces

- The PDF Generator is a subsystem that is responsible for generating PDF files based on the latest information. It is used by the GitHub server to generate PDF files based on the latest information.
- Google Maps is an external component that is used by the frontend to display maps of affected areas and help users locate nearby emergency resources.
- AWS is an external component that is used by the frontend to store and retrieve data. It is also used by the PDF Generator to store and retrieve data.

4.2.3 Internal Interfaces

As it can be observed from the figure above, afetbilgi.com has multiple internal interfaces. Frontend, PDF Generator, Maps and Data. The operations given in the diagram can be summarized as follows:

Operation	**Description**
redirectTo()	Users can access the website through a web browser or mobile device.
citySelection()	Users can filter the information based on the city they are in.
sendRequest()	Users can send requests to the system to fetch pieces of information.
changeLanguage()	Users can change the language of the website.
parseData()	Automated scripts hosted on GitHub servers parse the data stored on AWS.
parseMapData()	Automated scripts hosted on GitHub servers parse the map data stored on AWS.
readGoogleSheetsDoc()	Automated scripts hosted on GitHub servers read data from Google Sheets.
writeGoogleSheetsDoc()	Automated scripts hosted on GitHub servers write data to Google Sheets.
getCurrentPosition()	The system can get the current position of the user.
search()	The user can search for a keyword.
createLeafTemporaryAccomodationPDF()	The system can create a PDF file for temporary accomodation.
createLeafSafeGatheringPlacesPDF()	The system can create a PDF file for safe gathering places.
createLeafMealPDF()	The system can create a PDF file for meal information.
createPhoneNumbersPDF()	The system can create a PDF file for phone numbers.
createWebSitesPDF()	The system can create a PDF file for websites.
createArticlePDF()	The system can create a PDF file for articles.
createVeterinerPlacesPDF()	The system can create a PDF file for veteriner places.
createHelpCentersPDF()	The system can create a PDF file for help centers.
createStemCellsPDF()	The system can create a PDF file for stem cells.
createPharmacyPDF()	The system can create a PDF file for pharmacies.

Table 2: Operations on Internal Interfaces

4.2.4 Interaction Patterns

4.3 Information View

4.3.1 Stakeholders' Uses

There are four main stakeholders of the system: the users, emergency responders, information providers and the programmers. The users use this view to understand the data that is stored in the system. The emergency responders

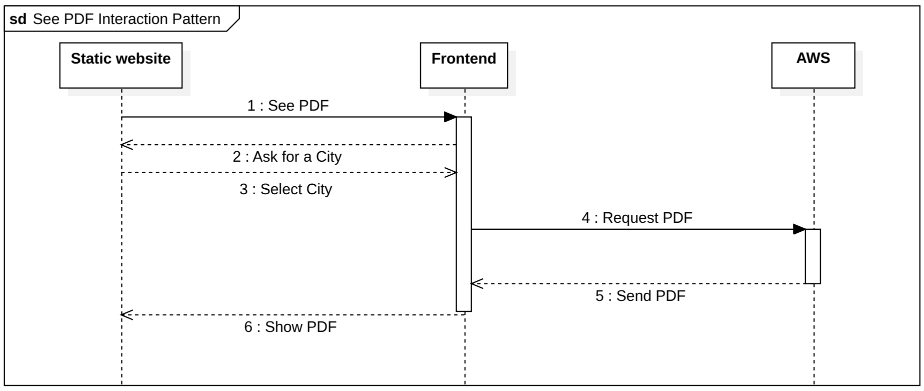

Figure 7: Sequence Diagram for Seeing PDF

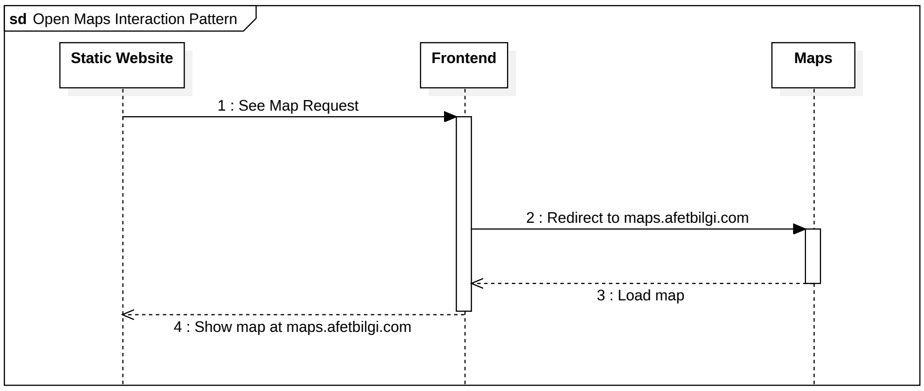
sd Open Maps Interaction Pattern
Static Website
Frontend
Maps
1 : See Map Request
2 : Redirect to maps.afetbilgi.com
3 : Load map
4 : Show map at maps.afetbilgi.com

Figure 8: Sequence Diagram for Opening Map

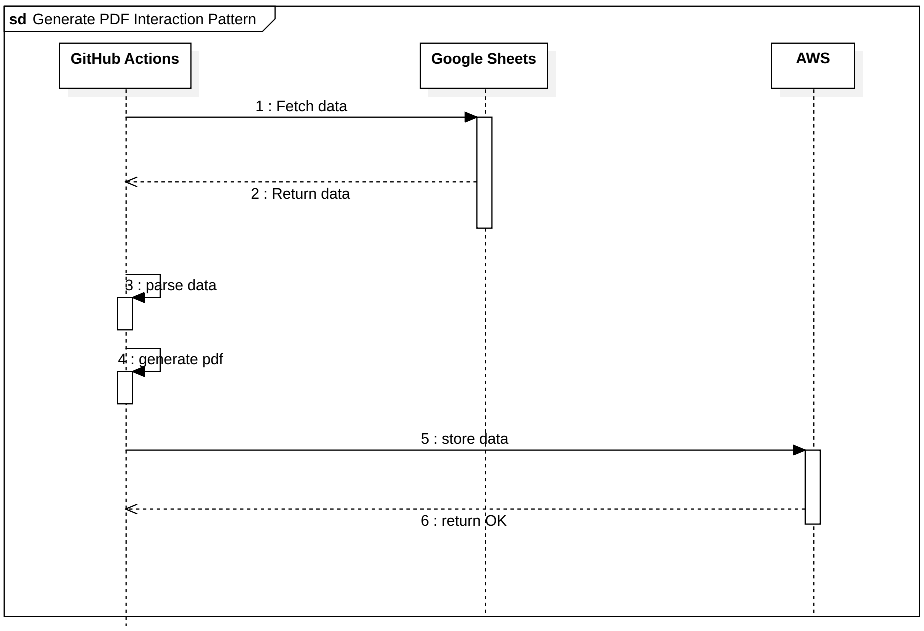

Figure 9: Sequence Diagram for Generating PDF

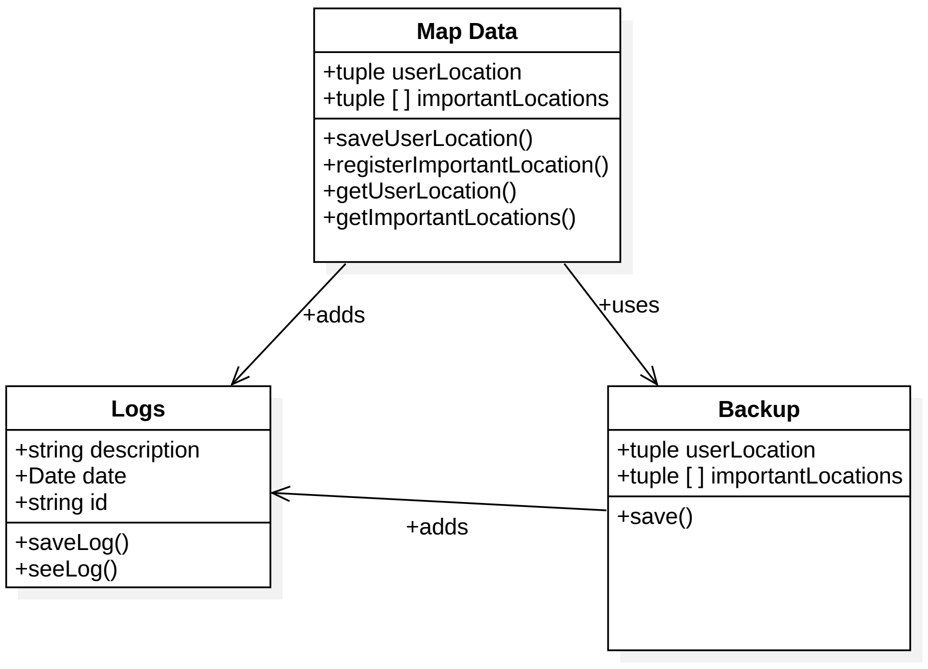

Figure 10: Database Class Diagram

use it to understand the data that is stored in the system to plan their response to the disaster. Information providers use it to understand what data is stored in the system and how it is used. Programmers use it to understand the data that is stored in the system to develop and maintain it.

4.3.2 Database Class Diagram

4.3.3 Operations on Data

- **saveUserLocation()** Saves the user's location to the database.
- **getUserLocation()** Retrieves the user's location from the database.
- **registerImportantLocation()** Registers an important location to the database.
- **getImportantLocations()** Retrieves important locations from the database.
- **saveLog()** Saves a log to the database.
- **seeLog()** Retrieves a log from the database.

- **save()** Saves the current state of the database to the backup bucket on AWS cloud.

4.4 Deployment View

4.4.1 Stakeholders' Uses

There are four main stakeholders of the system: the users, emergency responders, information providers and the programmers. The users use this view to understand how the system is deployed and how it reacts to changes, in other words, how dynamic it is. The emergency responders use it to understand how the system is deployed and how it reacts to changes to plan their response to the disaster. Information providers use it to understand how the system is deployed and how the data is processed and the changes are reflected. Programmers use it to understand how the system is deployed and how it reacts to changes to develop and maintain it.

4.4.2 Deployment Diagram

In this Deployment diagram, the system's deployment environment is shown. There are four main characters that play a role in the deployment of the system. These are GitHub, AWS, cloudflare and afetbilgi.com. A series of actions are performed by these characters to deploy the system. These actions are created by the developers and are hosted on GitHub. These actions interact with AWS and cloudflare to deploy the system. The system is deployed on AWS and cloudflare. Finally, the system is deployed on afetbilgi.com.

4.5 Design Rationale

4.5.1 Context View

The rationale behind the context view is to provide a high-level overview of how afetbilgi.com interacts with its external environment and stakeholders. This information is useful for understanding the scope of the system and identifying

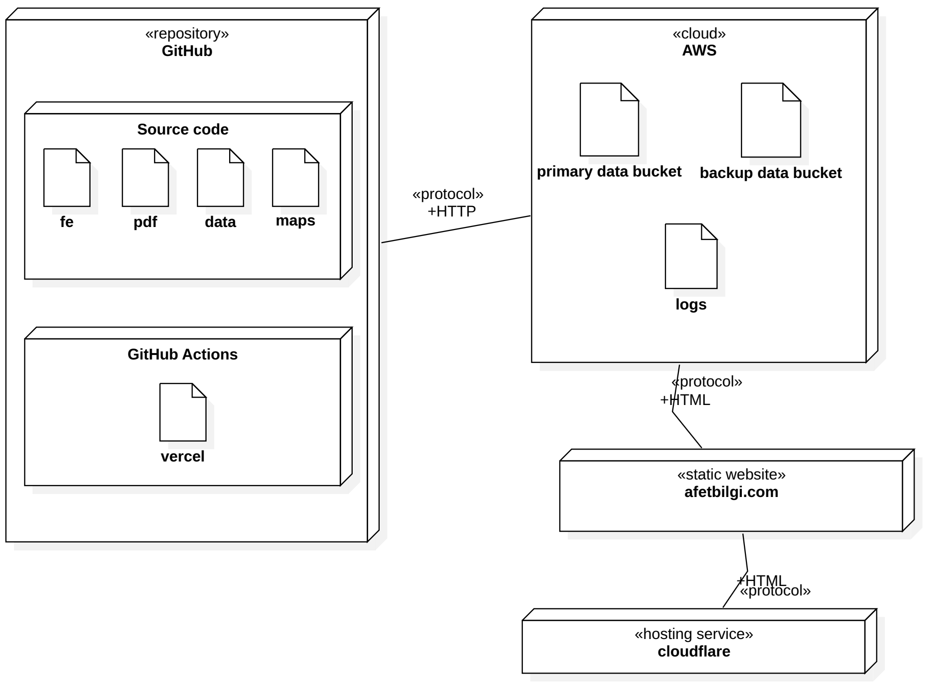

Figure 11: Deployment Diagram

potential risks or dependencies that may impact its operation.

4.5.2 Functional View

The rationale behind the functional view is to provide a detailed description of the capabilities and limitations of the system. This information is useful for understanding how the system can be used to respond to the disaster.

4.5.3 Information View

The rationale behind the information view is to provide a detailed description of the data that is stored in the system. This information is useful for understanding what data is stored in the system and how it is used.

4.5.4 Deployment View

The rationale behind the deployment view is to provide a detailed description of how the system is deployed and how it reacts to changes. This information is useful for understanding how dynamic the system is.

5 Architectural Views for Suggestions to Improve the Existing System

5.1 Context View

5.1.1 Stakeholders' Uses

There are four main stakeholders of the system: the users, emergency responders, information providers and the programmers. The users use this view to understand how the system is deployed and how it reacts to changes, in other words, how dynamic it is. The emergency responders use it to understand how the system is deployed and how it reacts to changes to plan their response to the disaster. Information providers use it to understand how the system is deployed and how the data is processed and the changes are reflected. Programmers use it to understand how the system is deployed and how it reacts to changes to develop and maintain it.

5.1.2 Context Diagram

As shown in the diagram, users can access the website through a web browser or mobile device. They can view information about natural disasters, emergency response procedures, and other related content. Administrators have additional privileges and can manage user accounts, update content on the website, and monitor site analytics.

Third-party services are also used to provide additional functionality on the website. For example, a map API is used to display maps of affected areas and help users locate nearby emergency resources.

Overall, the Context Diagram provides a high-level overview of how afetbilgi.com interacts with its external environment and stakeholders. This information is useful for understanding the scope of the system and identifying potential risks or dependencies that may impact its operation.

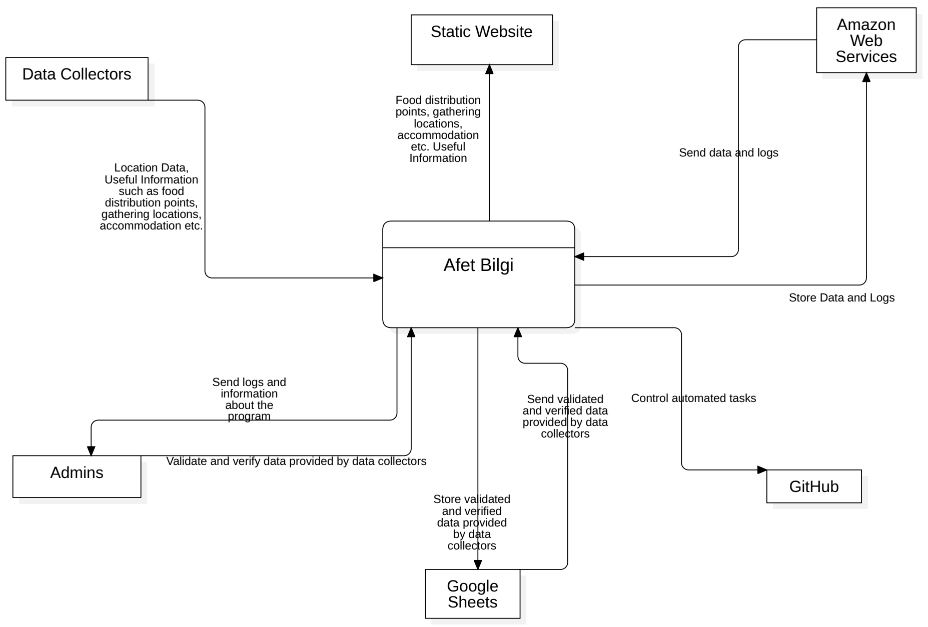
Static Website
Amazon Web Services
Data Collectors
Food distribution points, gathering locations, accommodation etc. Useful Information
Send data and logs
Location Data, Useful Information such as food distribution points, gathering locations, accommodation etc.
Afet Bilgi
Store Data and Logs
Send logs and information about the program
Send validated and verified data provided by data collectors
Control automated tasks
Admins
Validate and verify data provided by data collectors
GitHub
Store validated and verified data provided by data collectors
Google Sheets

Figure 12: Context Diagram

5.1.3 External Interfaces

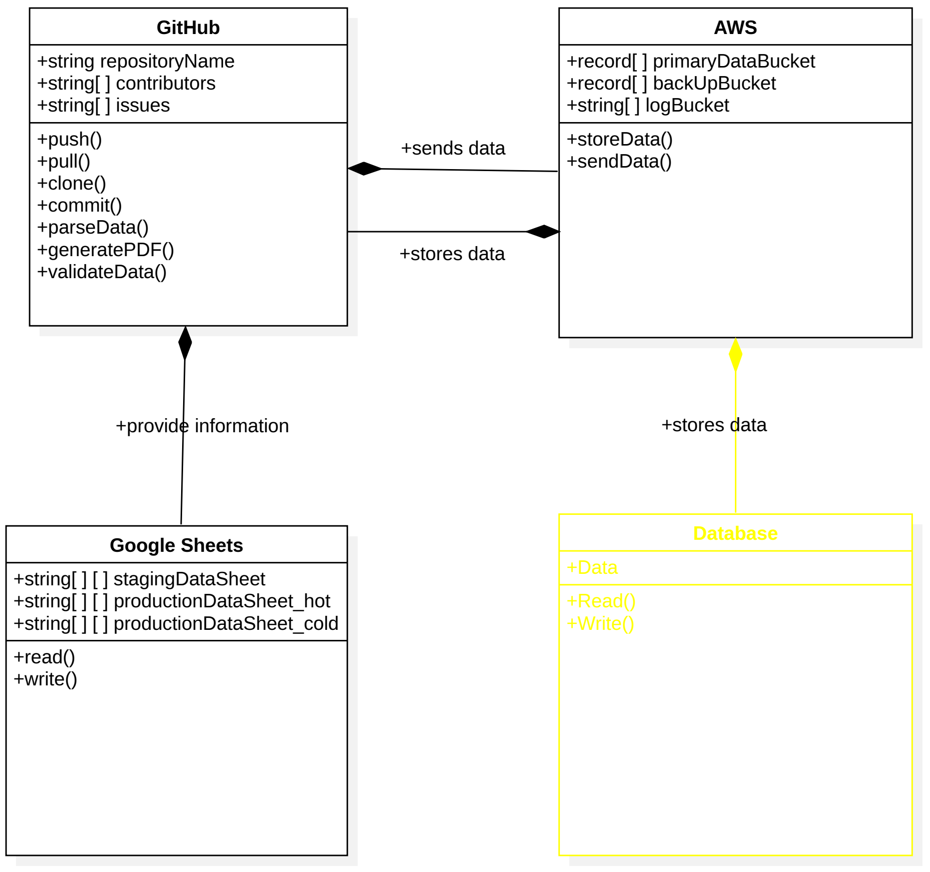

Figure 13: External Interfaces

A database component is used to store information about natural disasters, emergency response procedures, and other related content. The database is accessed through a web service that provides an interface for reading and writing data. The web service is hosted on a cloud platform and can be accessed from anywhere in the world.

Operation	**Description**
read()	Read data from the database
write()	Write (or delete) data to the database

Table 3: Operations on External Interfaces

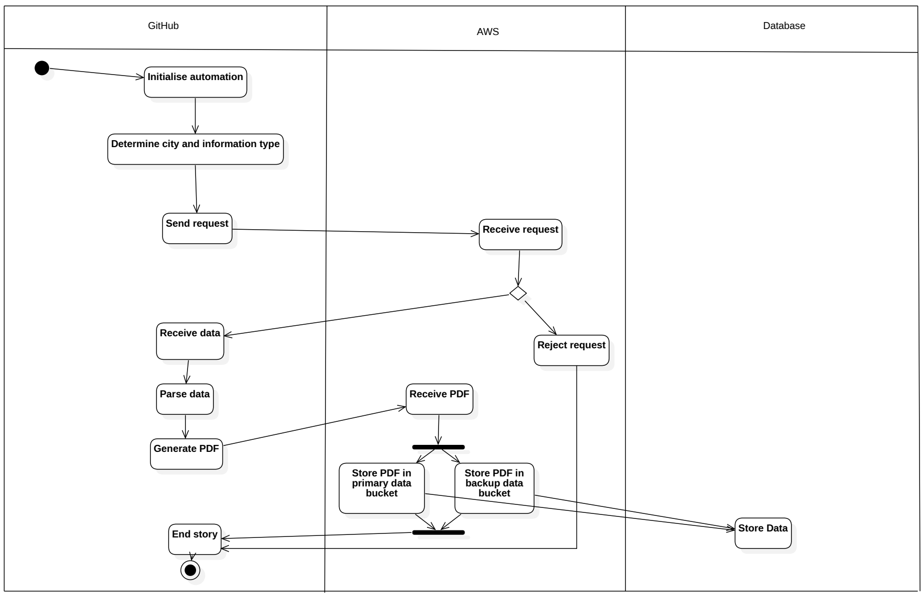

Figure 14: Activity Diagram for GitHub - AWS Interactions, Database Added

5.1.4 Interaction Scenarios

5.2 Functional View

5.2.1 Stakeholders' Uses

There are four main stakeholders of the system for suggestions to improve the existing system: the users, emergency responders, information providers and the programmers. The users use this view to understand the capabilities and limitations of the system for suggestions to improve the existing system. The emergency responders use it to understand the capabilities and limitations of the system for suggestions to improve the existing system to plan their response to the disaster. Information providers use it to understand the capabilities and limitations of the system for suggestions to improve the existing system and how the data is processed and the changes are reflected. Programmers use it to understand the capabilities and limitations of the system for suggestions to improve the existing system to develop and maintain it.

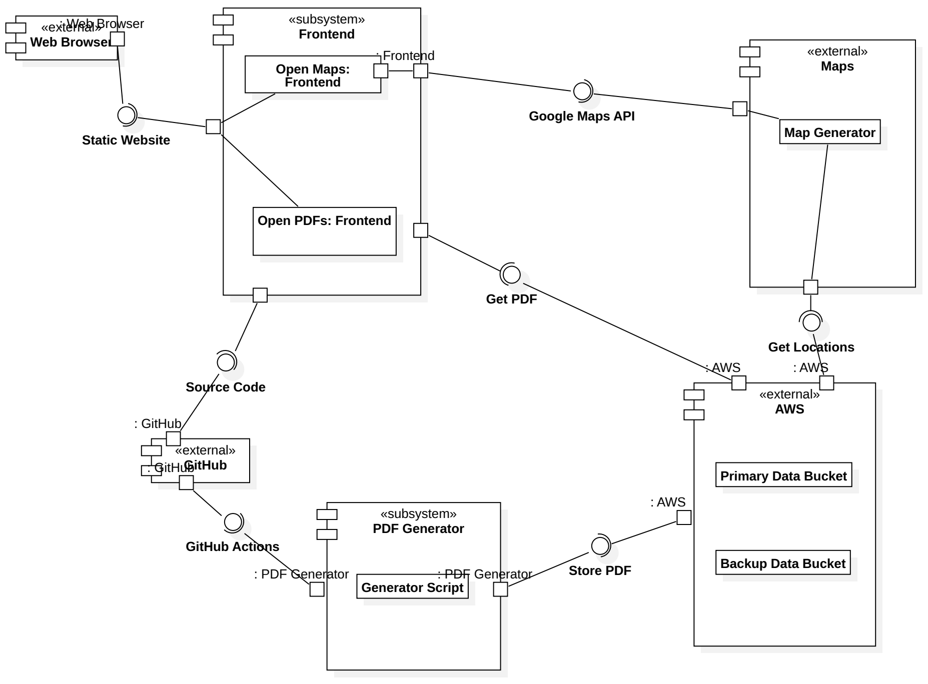

Figure 15: Component Diagram

5.2.2 Component Diagram

Our system consists of three main components: a web application, a database, and a web service. The web application is used by users to view information about natural disasters, emergency response procedures, and other related content. The database is used to store this information. The web service provides an interface for reading and writing data to the database.

5.2.3 Internal Interfaces

5.2.4 Interaction Patterns

5.3 Information View

5.3.1 Stakeholders' Uses

There are four main stakeholders of the system for suggestions to improve the existing system: the users, emergency responders, information providers and the programmers. The users use this view to understand the data that

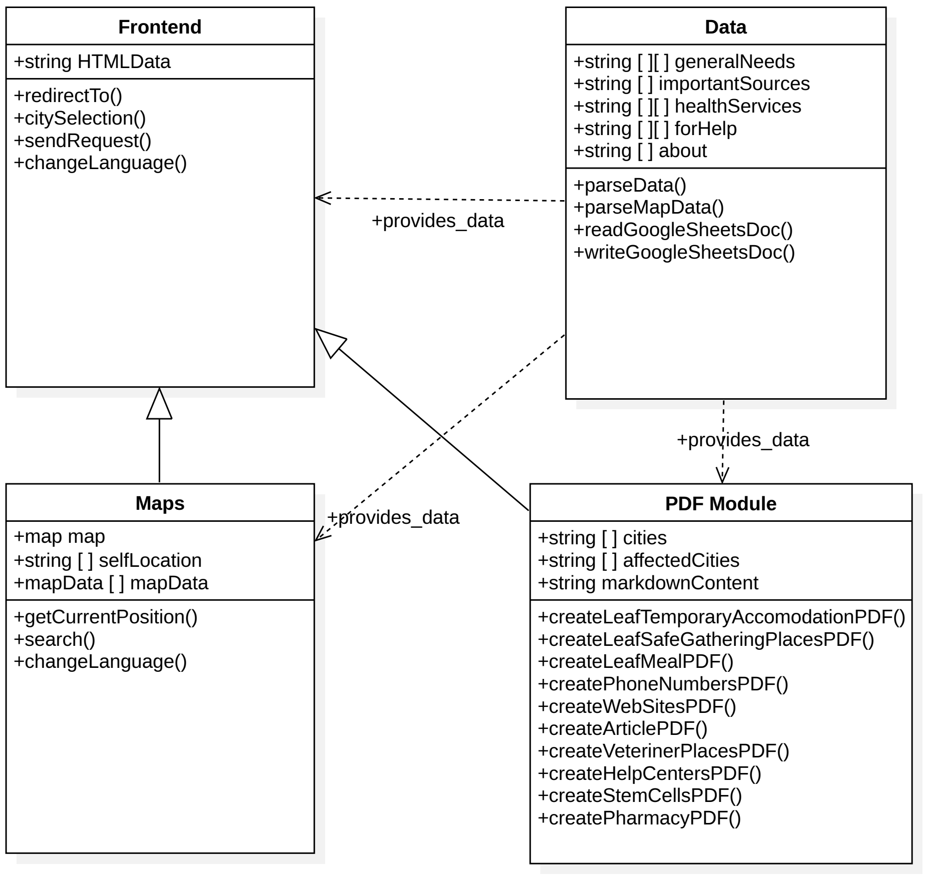

Figure 16: Internal Interfaces

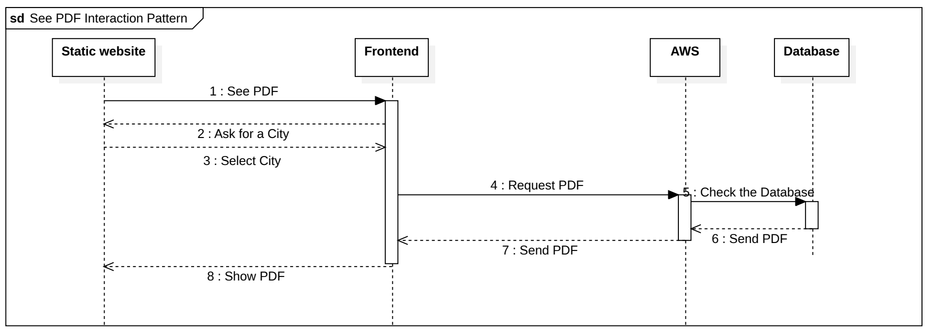

Figure 17: Sequence Diagram

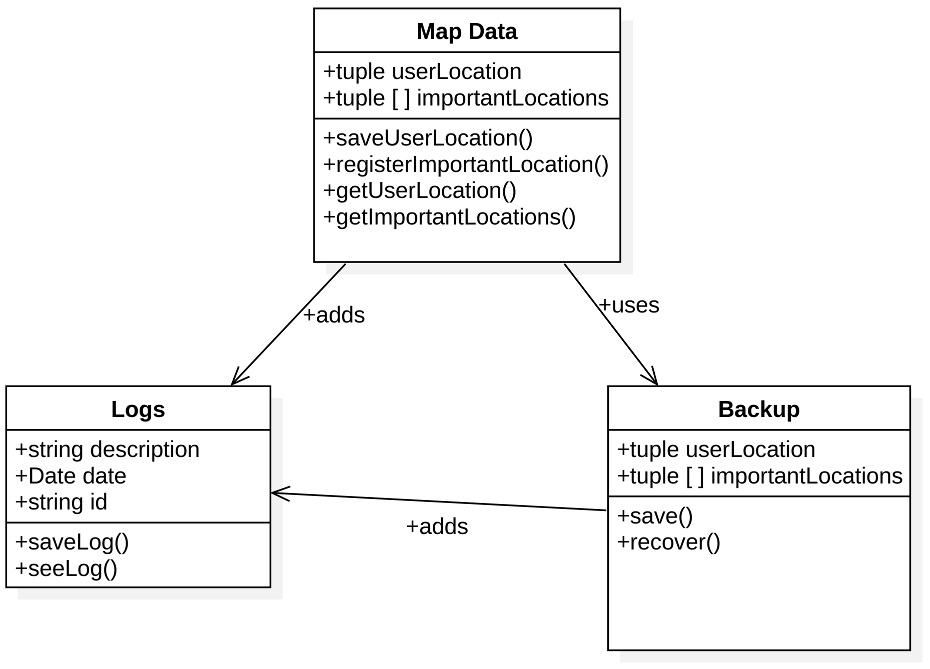

Figure 18: Database Class Diagram

is stored in the system for suggestions to improve the existing system. The emergency responders use it to understand the data that is stored in the system for suggestions to improve the existing system to plan their response to the disaster. Information providers use it to understand the data that is stored in the system for suggestions to improve the existing system and how the data is processed and the changes are reflected. Programmers use it to understand the data that is stored in the system for suggestions to improve the existing system to develop and maintain it.

5.3.2 Database Class Diagram

5.3.3 Operations on Data

Operation	**Description**
create()	Update data in the database
read()	Read data from the database
update()	Update data in the database
delete()	Delete data from the database

Table 4: Operations on Data

5.4 Deployment View

5.4.1 Stakeholders' Uses

There are four main stakeholders of the system for suggestions to improve the existing system: the users, emergency responders, information providers and the programmers. The users use this view to understand how the system is deployed and how it reacts to changes for suggestions to improve the existing system. The emergency responders use it to understand how the system is deployed and how it reacts to changes for suggestions to improve the existing system to plan their response to the disaster. Information providers use it to understand how the system is deployed and how it reacts to changes for suggestions to improve the existing system and how the data is processed and the changes are reflected. Programmers use it to understand how the system is deployed and how it reacts to changes for suggestions to improve the existing system to develop and maintain it.

5.4.2 Deployment Diagram

5.5 Design Rationale

5.5.1 Context View

The rationale behind the context view is to provide a high-level description of the system and its environment. This information is useful for understanding how the system interacts with its environment.

5.5.2 Functional View

The rationale behind the functional view is to provide a detailed description of the system's functionality. This information is useful for understanding what the system does and how it does it.

5.5.3 Information View

The rationale behind the information view is to provide a detailed description of the system's data. This information is useful for understanding what data the system stores and how it is stored.

5.5.4 Deployment View

The rationale behind the deployment view is to provide a detailed description of the system's deployment. This information is useful for understanding how the system is deployed and how it reacts to changes.