System Requirements

Specifications and Technical Design

**1.1 Plan to Address Objectives**

Our app will be built with anonymous and secure reporting in mind, enabling end-to-end encryption so that sensitive information cannot be altered or accessed. A stealth mode will be incorporated to further enhance user safety, allowing the application to appear as a neutral application, such as a calculator, reducing the risk of a perpetrator finding the app on a victim’s phone. This enables those affected by GBV to report with protection of identify and safety in mind.

With improvement ofaccessibility in reporting in mind, the application will be developed as a mobile-first application as smartphones are widely used across South Africa and ensures that incidents can be reported in real-time from any location. Since internet connectivity is less accessible in rural or less developed areas, an offline-first feature will be included in the application to allow reports to be saved locally on a user’s device and synced automatically once the user has connectivity again.

To develop emergency response through location information, GPS tagging will be integrated into the application to record the location of reported cases. This kind of data can be aggregated for NGOs and authorities to be able to map out high-risk areas in real time. Additionally, an SOS panic button will be included in the application’s features to allow survivors to instantly alert chosen contacts and support services nearby, providing their location for response.

A directory of certified NGOs, shelters, medical centers, and legal aid providers will be a provided feature in the application for survivors to access and locate the nearest available aid. Furthermore, the application will have a secure messaging capability which connects the survivors with service providers without revealing any sensitive, personally identifiable information – enabling trust and confidentiality in communication.

Lastly, we want to be as inclusive as possible, and since there are many languages spoken in South Africa, our application should have a multilingual interface. This would include widely spoken languages such as Zulu, isiXhosa, and English to start with, and possibly expand over time. There are also users with limited literacy, which can be addressed in our application through our voice messaging feature, enabling survivors to record their reports without having to type or read complex text.

**1.2 Project scope**

**1.2.1 Information scope**

Our application will gather and save GBV report data - this could be descriptions, evidence (pictures or videos), location data, and emergency contact information. Sensitive information will first be saved locally on the user's device for offline use and then synced to a safe cloud-based database when connectivity is available. Stored data will be end-to-end encrypted for user safety, and no sensitive information will be distributed without user consent. Confidentiality will be implemented by using authentication mechanisms in order for stored data to be accessed.

**1.2.2 Functional scope**

Our system will include features such as:

* Anonymous reporting with end-to-end encryption and an optional stealth mode.
* Offline reporting with automatic syncing when connectivity to the internet is recovered.
* Location tagging to assist in the mapping of high-risk areas.
* SOS button for instantaneous alerts to chosen contacts or support services.
* Directory of services such as NGOs, shelters, legal aid, and medical support.
* Secure messaging where survivors can speak to verified service providers.
* Voice messaging reporting for survivors with limited literacy.
* Multilingual support for languages such as Zulu, isiXhosa, and English (with future expansion).

**1.2.3 Communication scope**

In-app notifications will be used to communicate with the user, such as confirming a successfully submitted report. To enable accessibility in all areas, our app will be able to function on Wi-Fi and mobile data. An SOS panic button will be available in the case of emergencies to provide instant communication and response from the closest support services as well as trusted contacts chosen by the user. To improve privacy for the user, our app will have secure communication which will take place between the survivor and the NGOs through encrypted messaging rather than SMS.

**1.3 Business requirements**

Our GBV Reporting application has two main end-user groups, each with their own specific needs from the system:

1. Survivors:

GBV survivors need a safe, reliable, and user-friendly system. A survivor needs an app that does not put them at risk, allowing anonymous and secure reporting with a simple interface. Features like offline reporting, multilingual support, instant emergency aid through an SOS button, and stealth mode that disguises the app should be incorporated to meet user needs.

1. Support Providers (NGOs, legal aid, medical services, shelters):

Support providers need secure access to survivor reports in a way that does not expose sensitive survivor information unnecessarily. They need to be able to communicate securely through messages with a survivor, recognize high-risk areas based on GPS data, and they require a central database of reports. The application’s interface should be simple, dependable, and be able to support collaboration across different types of services.

**1.4 Hardware and Software Requirements**

State which hardware and software tools will be used both in developing the application and in its use, e.g:

**1.4.1 Software Requirements**

* The mobile application will be developed using Android Studio.
* The web application will be developed using Microsoft Visual Studio 2022.
* The mobile application will use Sqlite as its local storage medium.
* Microsoft SQL Server 2022 will be used as the shared online database.
* Connectivity between the app, the web interface and the shared database will be facilitated by developing a WCF Web service in Microsoft Visual Studio 2022.
* The mobile application requires a phone running (at least) Android 12 (Snow Cone).
* The web application will run in any browser but will be optimized for Google Chrome.

**1.4.2 Hardware Requirements**

* The mobile application requires a mobile phone with at least 1Gb or RAM, capable of running Android 12.
* The web application may be opened on any device with a web browser.
* The web application and SQL Server will be hosted on a shared web server.

**1.5 Design constraints**

In this section, discuss any constraints that there might be on the system development and / or deployment process. This can include things such as security, interface and performance constraints.

**1.5.1 Security constraints**

Discuss any issues related to the security of the proposed system, e.g.

The notes taken by the healthcare workers might be of a sensitive nature, as they contain information regarding patients seen by the healthcare workers. Therefore, care must be taken to encrypt the notes both on the phone and the server and to ensure that users should provide a username and password when accessing the system. The username and password should also be encrypted.

**1.5.2 Interface constraints**

Discuss any issue related to the interface of the proposed system, e.g.

The mobile interface needs to be user-friendly as it may be used by health care workers who are not technically savvy. The web interface should be responsive so that users have the choice to view the pages on a desktop, tablet or phone.

**1.5.3 Performance constraints**

The system should be optimized to ensure that searches through the web interface are fast and efficient. The mobile application should be optimized for use on low end mobile devices with low memory capacity and older processors.

**1.6 High-level use case diagram**

The purpose of a use case diagram in UML is to demonstrate the different ways that a user might interact with a system. A use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system.

***Example:***

Figure 1.1 presents a high-level use case diagram of the proposed system and its users.

A screenshot of a cell phone

Description automatically generated

Figure .1 High-level use case diagram

**1.7 UML Class Model / Diagram**

This section should present your (planned) UML class model. If you have not yet started your system development, you may draw the diagram using a stand-alone tool, such as Visio. Alternatively, if you have already started development, you may use the class diagram created for you by tools such as Visual Studio. This does not need to be the final version of your UML class model as you’ll be able to update the diagram in your final, combined report at the end of the semester. The number of classes in your diagram tends to align with the number of tables in your database.

***Example:***

Figure 1.2 represents the UML class diagram of the YOURPLANNEDSYSTEMNAME system.

A screenshot of a cell phone

Description automatically generated

Figure .2 UML Class Diagram

**1.8 Relational Database Model Diagram**

This section should present the (planned) version of your relational database model. If you have not yet started designing your database, you may draw the diagram using a stand-alone tool, such as Visio. Alternatively, if you have already started the design, you may use the ERD created for you by tools such as SQL Server Management Studio. This does not need to be the final version of your relational database model as you’ll be able to update the diagram in your final, combined report at the end of the semester. Remember, it needs to have the absolute minimum of the equivalent of 10 tables. Groups of three students will need at least 14 tables.

***Example:***

Figure 1.3 represents the entity relationship diagram of the YOURPLANNEDSYSTEMNAME system.

Diagram

Description automatically generated

Figure 1.3 ER Diagram

**1.9 User Interface Design**

This section should present the user interface designs of at least 8 screens from your proposed system (10 for groups of 3). If you have a system which is more graphically driven (like a game), you can substitute screens with the designs of individual components. If you have not yet started on your system development, you may use stand-alone tools such as Visio to create simple versions of your planned screens. These can be in the form of low fidelity wireframes. Alternatively, if you have already started system development, you may take screenshots of the screens you have already completed. For each screen(or component) you are required to provide a description of the functionality. These do not need to be the final versions of these screens as you’ll be able to update the screens in the final, combined report at the end of the semester.

***Example:***

Figure 1.4 represents the login screen of the YOURPLANNEDSYSTEMNAME system. The screen requires that a user enter their email address (as a username) and a password. Upon successful login, the user will be redirected to the system’s landing page. The screen also provides the user with an option to use in the event that they have forgotten their password.

A screenshot of a cell phone

Description automatically generated

Figure 1.4 Login Screen