**Chapter III**

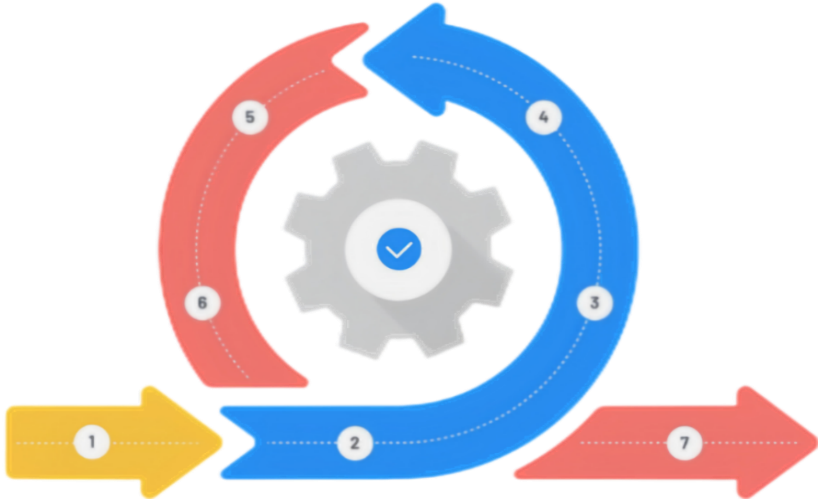
In this chapter, we explain how we'll build our Smart Road Pothole and Traffic Hazard Mapping website. We're taking a careful step-by-step approach to make sure the website works well for reporting and fixing road problems. This chapter shows exactly how we'll develop the website to help DPWH fix potholes and help drivers avoid road hazards.

Our method starts by asking DPWH engineers, drivers, and commuters what they need through surveys and interviews. This helps us understand what the website should do - like how people should report potholes, how to rate how dangerous these road hazards are, and how to give drivers live directions that avoid bad road sections.

For our technology, we'll use HTML for page structure, CSS for styling, JavaScript for interactive elements, PHP for server processing, MySQL for the database, and both Leaflet and TomTom API for maps. Leaflet will help us create interactive maps that show potholes and road hazards, while TomTom API will give us traffic data and live navigation features to help drivers avoid dangerous areas in real-time.

After we build the website, we'll train DPWH maintenance teams and local officials how to use it. We'll also run campaigns to get drivers to report potholes and use the live navigation for safer travel. We'll keep watching how the website works, make updates, and improve it based on feedback from both DPWH and regular drivers.

**Agile Model Diagram**



**TESTING**

**DEPLOYMENT**

**PLANNING**

**REQUIREMENTS**

**MAINTINANCE**

**DEVELOPMENT**

**DESIGN**

**Figure 1. System Development Life Cycle**

We will choose the Agile model to build our Smart Road Pothole and Traffic Hazard Mapping website because it will let us get feedback often from DPWH engineers and drivers who report potholes. This way, we can make quick changes based on what users tell us about road hazards they find and how the live navigation is working.

By talking regularly with DPWH maintenance teams, district engineers, and everyday drivers, we will make sure the website helps identify potholes and road damage effectively while giving drivers good directions. Agile will let us test often and fix problems quickly, which is important for a website tracking constantly changing road conditions and providing live navigation.

This approach will help us build important features faster, like pothole alerts, road hazard maps, and live navigation that gives voice alerts and shows safe routes that avoid damaged roads. Agile is flexible and team-based, which makes it perfect for creating a website that helps DPWH repair roads better while also helping drivers avoid dangerous potholes and obstacles during their trips.

**PLANNING**

In the planning phase, we work closely with the Department of Public Works and Highways (DPWH) who will fix the roads, and also with regular drivers who will use our website to report problems and get safer directions. We make sure our website helps DPWH track different road problems - like potholes, flooded areas, things blocking the road, broken signs, and cracked pavement. At the same time, we create easy-to-use screens for drivers to report these problems and get live navigation with voice warnings when they're driving near dangerous spots. We organize our work to build both parts: web tools for DPWH to plan repairs and decide what to fix first, and a user-friendly website for drivers to take pictures of road problems, mark their exact location, and get turn-by-turn directions that avoid dangerous roads during their trips.

**REQUIREMENTS**

In this phase, we figure out what our website needs to do for DPWH road maintenance teams and regular drivers. We list key features like a map showing potholes, road cracks, and flooding, plus a live navigation system that gives drivers real-time directions to avoid these hazards. We make sure DPWH engineers can easily see which areas have the most severe road damage needing urgent repair, while drivers can both quickly report problems they find and also use the website's navigation feature that gives them voice directions like "caution, pothole ahead" or "turning right to avoid flooded area." To keep the website useful in real life, we get feedback from DPWH district offices about repair planning and from drivers about how the navigation works on their actual trips.

**DESIGN**

In the design phase, we create a website that works well for both DPWH and regular drivers. The back-end safely stores all pothole and hazard data including pictures, location, and severity ratings, while the front-end gives DPWH staff a dashboard showing all reported road problems on a map. For drivers, we design simple screens to report potholes with just a few taps and a live navigation view that works like Google Maps but with special warnings about road hazards. This navigation part will give turn-by-turn directions with both visual alerts on the screen and voice warnings like "pothole on right side in 100 meters" that help drivers avoid dangers without looking at their phones. We use TomTom API for the live navigation, making it recalculate routes automatically when new hazards are reported ahead of the driver's planned path.

**DEVELOPMENT**

During development, we build the main parts of our website bit by bit, starting with the pothole reporting tool and the live navigation system that helps drivers avoid hazards. The navigation works like a GPS app but shows special alerts for potholes and gives spoken warnings as drivers approach problem areas. We make sure this navigation works well even when internet connection is spotty by storing some map data on the user's device. We also build the DPWH dashboard that shows all road hazards on a map with color coding for severity. We focus on making data flow quickly from driver reports to both the DPWH repair teams and to other drivers who might be heading toward that same pothole or hazard, so the live navigation can update routes for everyone as soon as new problems are reported.

**TESTING**

In testing, we check that our website works well for both reporting problems and giving safe directions. We test the pothole reporting screen with regular people to make sure anyone can easily submit reports with photos. We test the live navigation by having drivers use it on actual trips, making sure the voice alerts work well and give enough warning time before reaching hazards. We check that the system correctly reroutes drivers when there's a serious hazard ahead and that the voice instructions are clear and not too distracting. For the DPWH side, we test how the dashboard handles multiple reports coming in at busy times and if the priority sorting helps them identify the worst potholes that need immediate fixing. We also make sure all parts of the website work on both computers and mobile phones since drivers will mostly use phones.

**DEPLOYMENT**

When we launch the website, we make it available to both DPWH offices and the public, with special focus on getting drivers to use the free live navigation feature that helps them avoid potholes and road hazards. We train DPWH maintenance teams how to use the dashboard for tracking road problems and create simple guides showing drivers how to both report hazards and use the turn-by-turn navigation with hazard alerts. We run ads on radio and social media explaining how the website gives better directions than regular map apps because it warns about potholes and bad road sections. We monitor how many people use the navigation feature, how many potholes get reported, and how quickly DPWH fixes them after they appear on our map. This helps us show how the website makes driving safer by warning about hazards while also helping get roads fixed faster.

**Fishbone Diagram**

Lack of archived reports on past road hazards.

Lack of real-time monitoring tools.

Inefficient reporting of new road hazards.

Unidentified specific road hazards and pothole locations in due to inefficient data collection and reporting.

Lack of Road Hazard Information

Limited Accessibility to Road Data

Failure to Identify New Incidents

Unrecorded Road Hazard History

Reports fail to specify exact locations.

Can’t access to accurate or complete information about the road

Difficulty obtaining real-time updates.

**Figure 2. Fishbone Diagram**

This fishbone diagram shows why it's hard to find and fix road problems like potholes. The big reason is the way people collect and share road info is not good. There are four main problems: people don’t report road issues well or forget to say where it is, there’s no fast way to check roads, there are no good tools to watch and save road data, and old road problems are not saved or shared. To fix this, we need a better system that helps people report road issues fast, shows the exact place, saves the info, and gives live updates. This will help make roads safer.

**Gantt Chart**

A Gantt Chart organizes tasks over a set timeline, with a time scale at the top and a list of activities on the side. Each task is represented by a bar, where the length and position indicate the start, progress, and completion of the activity. This visual tool simplifies planning and tracking progress.

**Definition**

Represented the months and their duration.

Represented the border of the activities.

Represented an activity that has already been done.

Represents a portion of an activity that has not been yet.

**Symbols**

Vertical line

Horizontal line

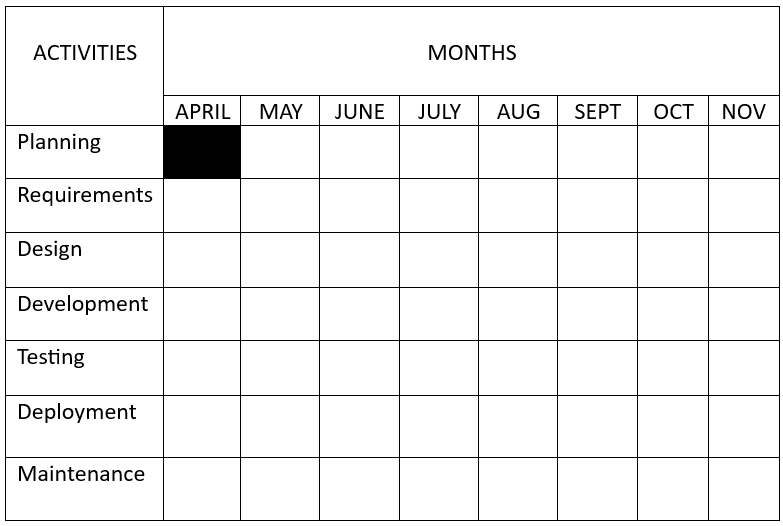


Shaded Box

Unshaded Box

**Gantt Chart Diagram**

The Gannt Chart will help the researcher plan out their steps in a span of time. The left side represents the list of activities while the top represents the time scale. Each activity is represented by the bar and the length of it represents the start date, duration and the end of the activity.



**Table 1.**

This timeline shows how we will make the hazard website. In April, we plan what the website will do. In May, we design how it looks and works. From June to September, we build the website. Users can report road problems, see hazards on the map, and use live navigation to stay safe. In October, we test everything to fix any bugs. In November, the website goes live. People can use it to report hazards, check the map, and get directions. We also ask users for ideas to make it better.

**Table 2.**

**Technical Feasibilities**

**Table 1. Hardware Specification (Development)**

Shows the hardware requirement needed in implementation of the project.

|  |  |
| --- | --- |
| **Hardware** | **Specifications** |
| Laptop | **Device name:** DELL INSPIRON  **Processor:** 11th Gen Intel(R) Core (TM) i3-1115G4 @ 3.00GHz 3.00 GHz  **Installed RAM:** 8.00 GB (7.74 GB usable)  **Device ID:** 3A2CF34F-7005-4F8C-8368-F99B3B04C9CF  **Product ID:** 00327-31077-50782-AAOEM  **System type:** 64-bit operating system, x64-based processor  **Pen and touch:** No pen or touch input is available for this display |

. **Table 3.**

**Table 2. Software Specification (Development)**

Shows the software requirement needed in implementation of the project.

|  |  |
| --- | --- |
| **Software** | **System Development Requirements** |
| Integrated Development Environment (IDE) | PHP, HTML, JavaScript, Bootstrap v5, Tailwinds, Visual Studio Code, CSS |
| API | TOM-TOM API |
| MAP | Leaflet map |
| Operating System | Windows 10, Windows 11 |
| Database | XAMPP (MySQL) |

**System Flowchart**

This flowchart shown physical control or resource level control. This flowchart shows process, system, or algorithm, using standardized symbols to represent the various steps and their sequence.

Process

Indicates any processing function.

**Symbols**

Terminator

**Description.**

Indicates the beginning or end of a program flow in the diagram.

On-page Reference

On-page connector is used to replace long line on a flowchart page.

Off-page Connector

An off-page connector is used when the target is on another page.

Data

Flow Line

Can represent any type of data in a flowchart may it be an input or output data.

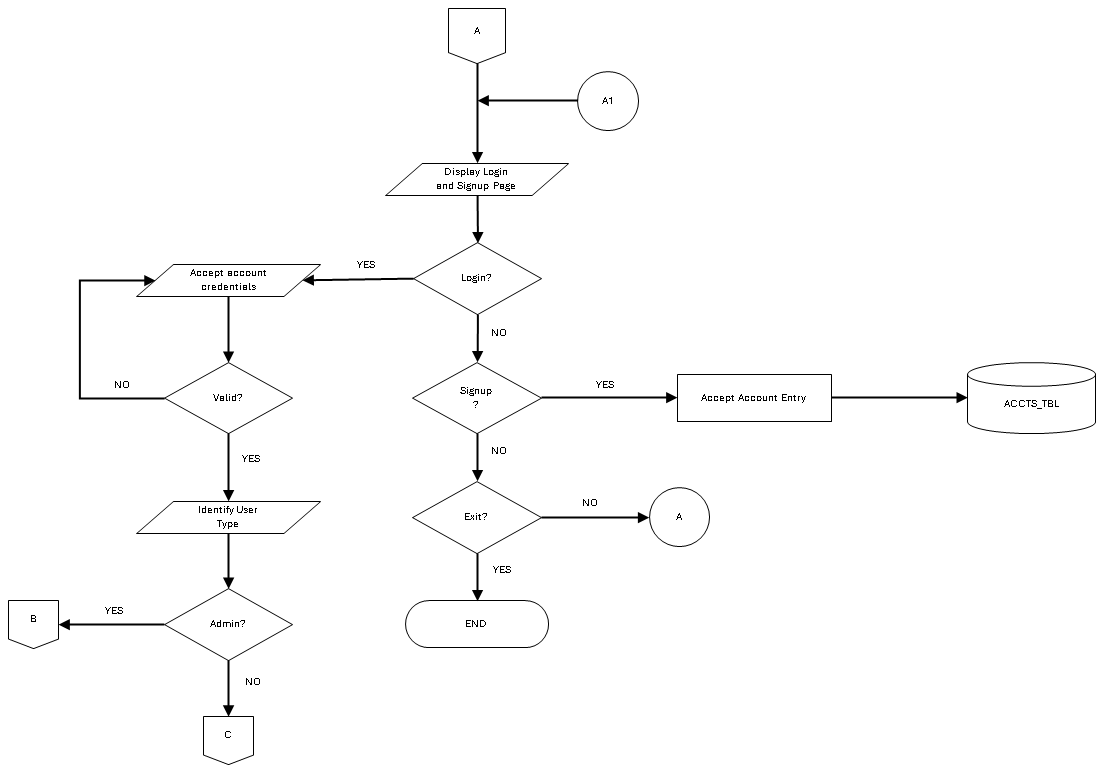
shows the process description.

Decision

Indicates a decision point between two or more paths in flowchart

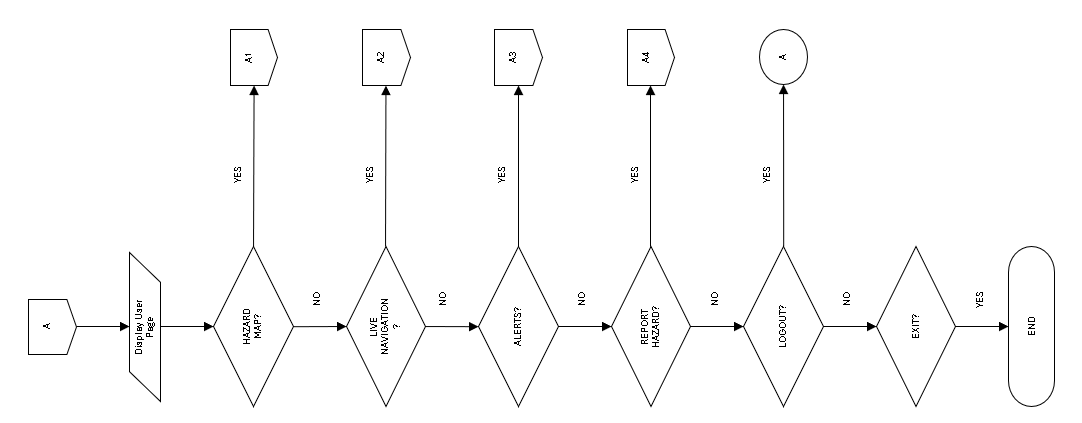
Database

Used For representing information kept in a modern storage database.

****

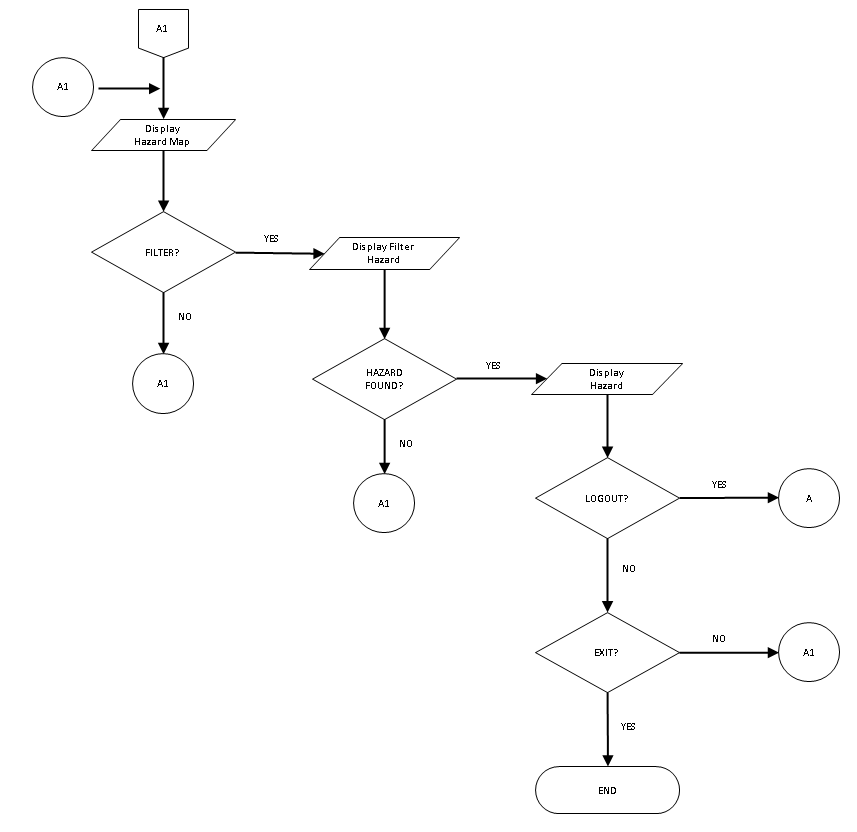
**Figure 3.** **User Authentication Flow**

This flowchart shows how users get into the Road Sense system. First, the system shows a login and signup page. If a person already has an account, they can login by entering their email and password. The system checks if these are correct - if they're wrong, it asks again. If they're right, the system figures out if they're an admin or a regular user. Admin users go to path B (the admin dashboard), while regular users go to path C (the user map page). If a person doesn't have an account, they can sign up for a new one, and their information gets saved in the accounts table. If they don't want to login or signup, they can exit and the system will end.

****

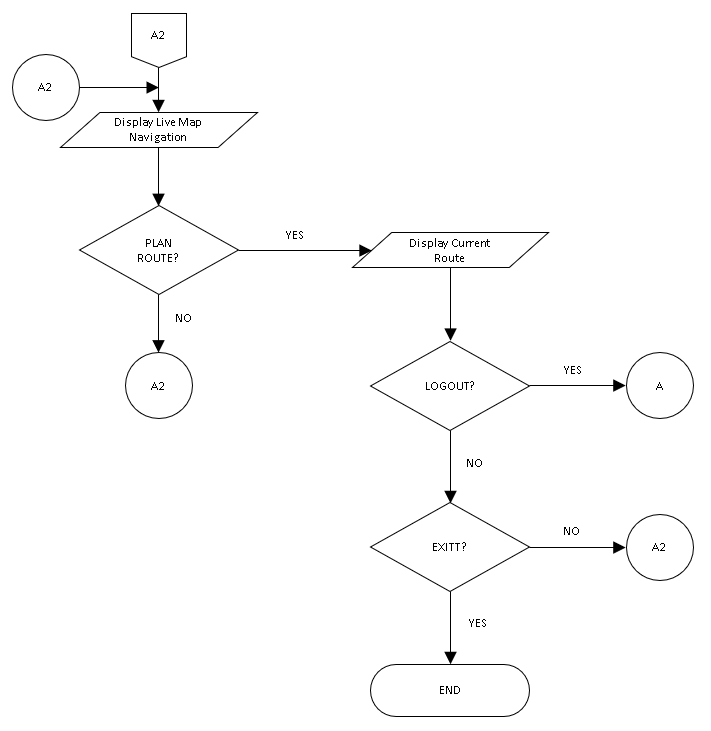
**Figure 4: User page**

shows the flowchart represents the navigation path for a user on the website. Each decision point offers the user a choice to navigate to a specific page (Hazard Map, Live Navigation Map, Alerts, Report Hazard, or Logout). If the user chooses "Yes" at any decision point, they are taken to the corresponding page (off-page navigation). If the user chooses "No," they continue to the next decision point until they either log out or exit the website.

****

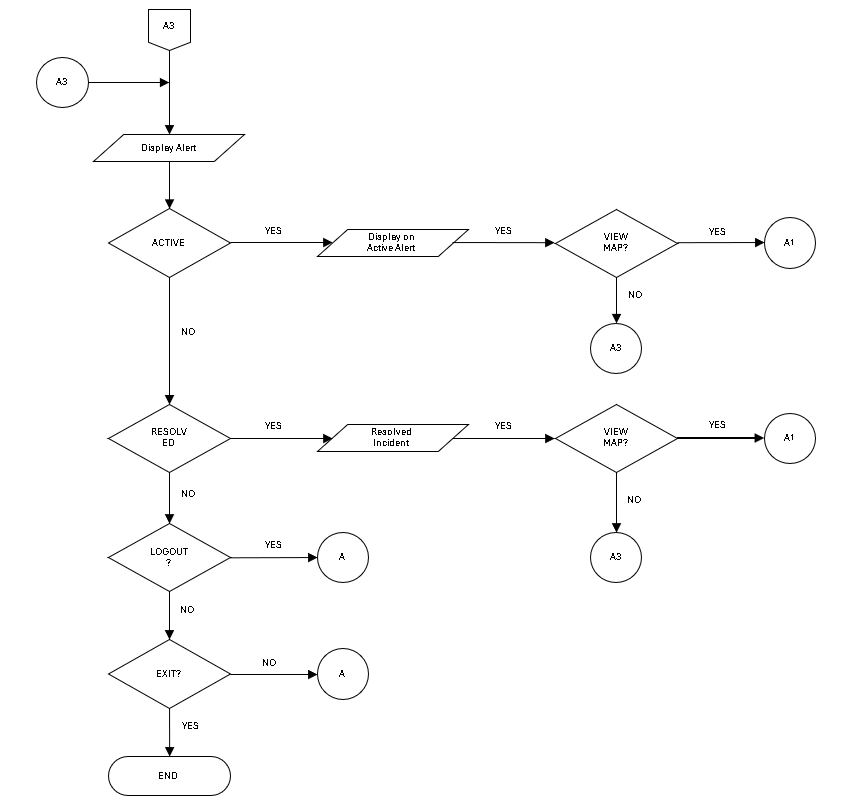
**Figure 5. Hazard map page**

The flowchart outlines the page for users within this specific section of the website. Users can choose to filter the hazards displayed on the map, view the filtered results, and decide whether to log out or exit the page. If hazards are found based on the filter criteria, they are displayed to the user; otherwise, the user is returned to the main Hazard Map page. This flowchart ensures that users can easily navigate and interact with the Hazard Map features.

****

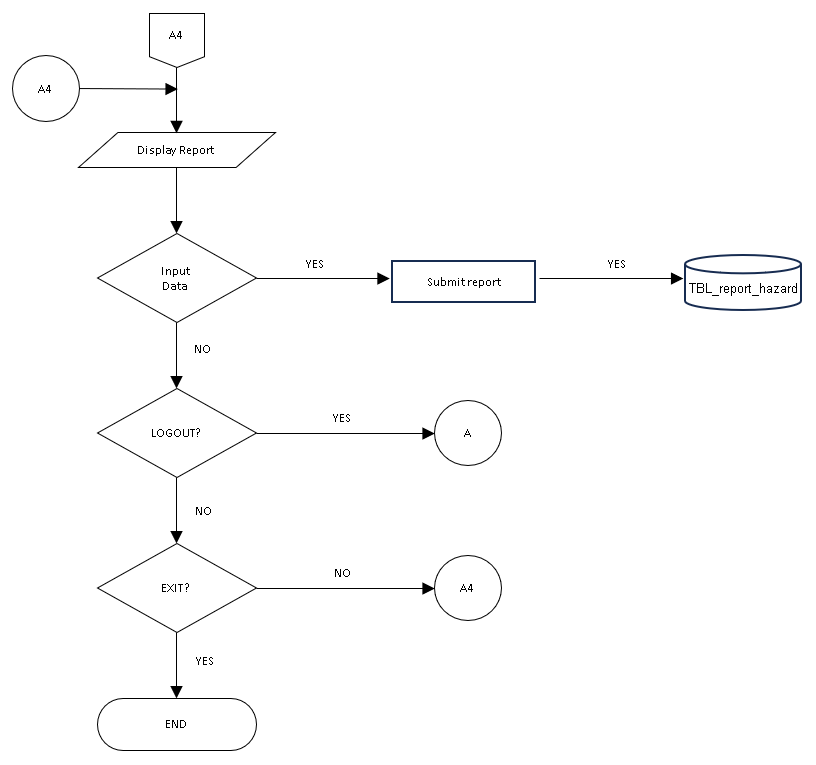
**Figure 6. Live navigation page**

The flowchart outlines the navigation path for users within this specific section of the website. Users can choose to plan a route and view the current route. At each decision point, users have the option to select "No," which will return them to the main Live Map Navigation page or keep them within the current flow without exiting or logging out.

****

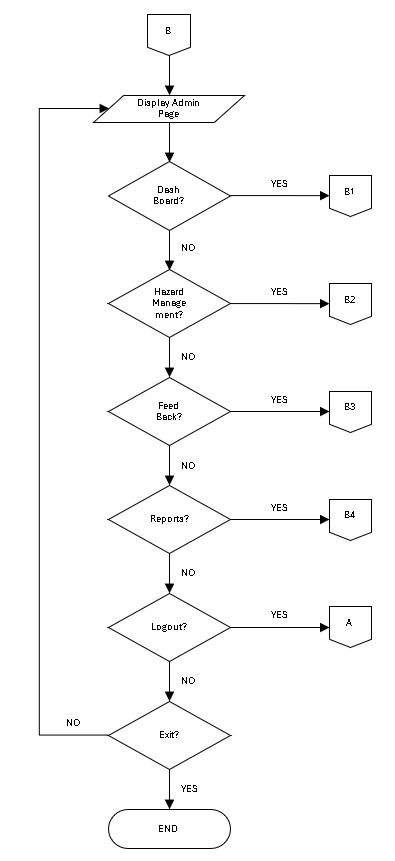
**Figure 7. Alert page**

The flowchart shows how users can see and interact with alerts. Starting at A3, users see alerts and can check if they are active or fixed. If an alert is active, they can view it on a map. Saying "Yes" takes them to the map, while "No" keeps them on the Alerts page. If an alert is fixed, they can also view it on a map. Users can log out to go back to the main page or exit to end the process. At each step, saying "No" keeps them on the same page.

****

**Figure 8. Report page**

The flowchart outlines the navigation path for users within this specific section of the website. Starting at point A4, users are presented with the Report page where they can input data to report a hazard. If data is inputted, the report is submitted and stored in the database. Throughout the process, users have the options to log out, which takes them back to the main user page (A), or exit the Report Hazard page, ending the process. At each decision point, users have the option to select "No," which keeps them within the current flow without exiting or logging out.

****

**Figure 9. Admin page**

The flowchart outlines the navigation path for admin users within this specific section of the website. Starting at point B, admin users are presented with the admin page where they can access various administrative functions such as the dashboard, hazard management, feedback, and reports. At each decision point, admin users have the option to select "No," which keeps them within the current flow without exiting or logging out.

B1

Display Dash Board

Exit?

END

B1

NO

YES

B1

**Figure 10. Dashboard page**

The flowchart outlines the navigation path for admin users within this specific section of the website. Starting at point B1, admin users are presented with the Admin Dashboard where they can view various metrics and information. At the decision point, admin users have the option to exit the Dashboard page. If they choose "Yes," the process ends. If they choose "No," they are taken back to the starting point "B1," allowing them to continue using the Dashboard.

B2

Display Hazard Management

Exit?

END

B2

NO

YES

B2

**Figure 11. Hazard management page**

The flowchart outlines the navigation path for admin users within this specific section of the website. Starting at point B2, admin users are presented with the Hazard Management page where they can manage various hazards. At the decision point, admin users have the option to exit the Hazard Management page. If they choose "Yes," the process ends. If they choose "No," they are taken back to the starting point "B2," allowing them to continue managing hazards.

B3

Display Feedback

Exit?

END

B3

NO

YES

B3

**Figure 11. Feed back page**

The flowchart outlines the navigation path for admin users within this specific section of the website. Starting at point B3, admin users are presented with the Feedback page where they can view and manage user feedback. At the decision point, admin users have the option to exit the Feedback page. If they choose "Yes," the process ends. If they choose "No," they are taken back to the starting point "B3," allowing them to continue managing feedback.

B4

Display Reports

Exit?

END

B4

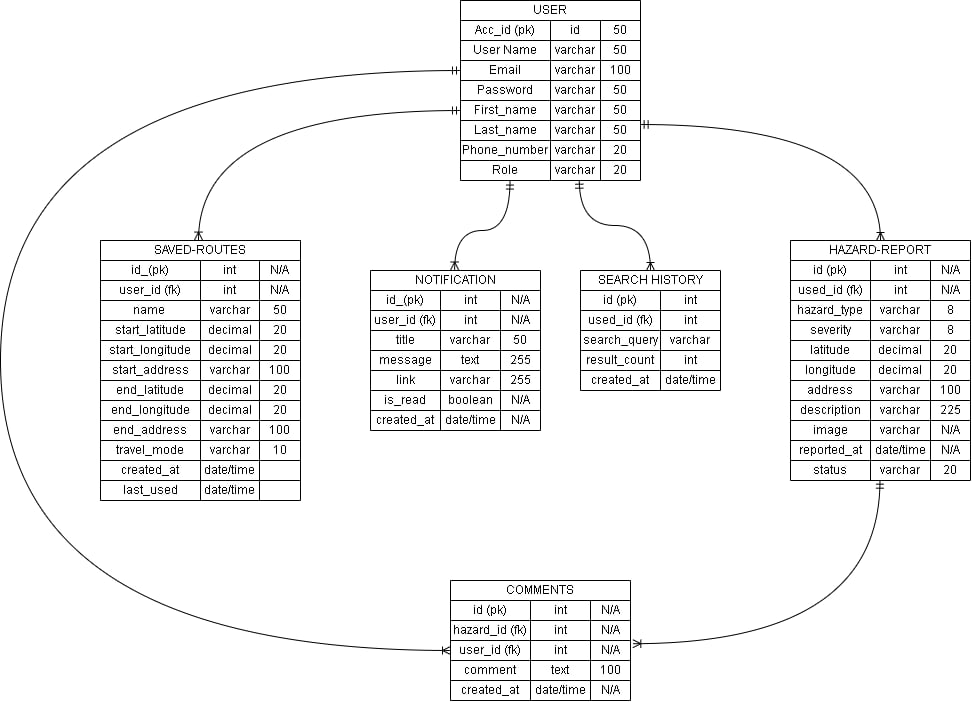
NO

YES

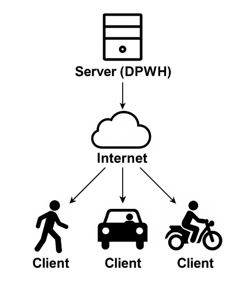
B4

**Figure 12. Reports page**

The flowchart outlines the navigation path for admin users within this specific section of the website. Starting at point B3, admin users are presented with the Reports page, where they can view all the reported hazards submitted by users. At the decision point, admin users have the option to exit the Report page. If they choose "Yes," the process ends. If they choose "No," they are taken back to the starting point “B3", allowing them to continue viewing the reports.



**Figure 13. Entity relationship diagram**

****

**Figure 14. System Architecture**

The figure shows the system architecture of the proposed Smart Road Pothole and Traffic Hazard Mapping Website, which uses a client-server model connected through the internet. On the server side, the DPWH (Department of Public Works and Highways) manages the backend system using desktop computers. They handle hazard reports, update the road hazard map, and monitor live road condition data. On the client side, users such as walkers, car drivers, and motorcycle riders access the system through their devices. They can report road hazards, view hazard maps, and use live navigation for safer travel. The server and clients communicate over the internet, allowing real-time reporting, map updates, and navigation support.

**Data Dictionary**

Data dictionary provide a precise vocabulary for specific data elements, helping to standardize a data set and ensure that the relevance and quality of data elements are consistent for all users. They describe the meaning and purpose of data elements within the context of a project and offer guidance on their interpretation.

|  |  |  |  |
| --- | --- | --- | --- |
| **TBL\_User\_Account** | | | |
| **Field Name** | **Field Type** | **Zise** | **Description** |
| Acc\_id (pk) | Int | N/A | Unique identifier for accounts |
| User\_name | varchar | 50 | User’s name of user |
| Email | Varchar | 50 | Login email of user |
| Password | Varchar | 100 | Login password of user |
| First\_name | Varchar | 50 | User’s first name |
| Last\_name | Varchar | 50 | User’s last name |
| Phone\_number | Varchar | 20 | User’s Contact number |
| Role | Varchar | 20 | Define the role or level of acces |

**Table 1. Data Dictionary of TBL\_User\_Account**

|  |  |  |  |
| --- | --- | --- | --- |
| **TBL\_Saved\_Route** | | | |
| **Field Name** | **Field Type** | **Zise** | **Description** |
| Id (pk) | Int | N/A | Unique identifier for saved route |
| User\_id (fk) | Int | N/A | Unique identifier for user that saved route |
| Name | varchar | 50 | Name of the user |
| Start\_latitude | Decimal | 20 | Start\_loaction of the user |
| Start\_longitude | Decimal | 20 | Start\_loaction of the user |
| Start\_address | Varchar | 100 | Start\_address of the user |
| End\_latitude | Decimal | 20 | Destination of the user |
| End\_longitude | Decimal | 20 | Destination of the user |
| End\_address | Varchar | 100 | Destination of the user |
| Travel\_mode | Varchar | 10 | Selection type of travel |
| Created\_at | Date/time | N/A | Date/time that start routing |
| Last\_used | Date/time | N/A | Date/time that end routing |

**Table 2. Data Dictionary of TBL\_Saved\_Route**

|  |  |  |  |
| --- | --- | --- | --- |
| **TBL\_Comments** | | | |
| **Field Name** | **Field Type** | **Zise** | **Description** |
| Id (pk) | Int | N/A | Unique identifier of each comment |
| Hazard\_id (fk) | Int | N/A | Unique identifier of the specific hazard |
| User\_id (fk) | Int | N/A | Unique identifier for the user |
| Comment | Varchar | 100 | Any kind of comments on the hazard |
| Created\_at | Date/time | N/A | Date/time that comments started |

**Table 3. Data Dictionary of TBL\_Comments**

|  |  |  |  |
| --- | --- | --- | --- |
| **TBL\_Hazard\_Report** | | | |
| **Field Name** | **Field Type** | **Zise** | **Description** |
| Id (pk) | Int | N/A | Unique identifier for each hazard report |
| User\_id (fk) | Int | N/A | Unique identifier for specific user that report hazard |
| Hazard\_type | varchar | 8 | Selection of what kind of hazard |
| Severity | Varchar | 8 | Selection of level of damage |
| Latitude | Decimal | 20 | Exact location of the reported hazard |
| Longitude | Decimal | 20 | Exact location of the reported hazard |
| Address | Varchar | 100 | Address of the location that hazard reported |
| Description | Varchar | 255 | Any explanation of the hazard reported |
| Image | Varchar | N/A | Image of the hazard that captured |
| Reported\_at | Date/time | N/A | Date/time that reported the hazard |
| status | Varchar | 20 | Status of the reported hazard |

**Table 4. Data Dictionary of TBL\_Hazard\_Reports**

|  |  |  |  |
| --- | --- | --- | --- |
| **Notification** | | | |
| **Field Name** | **Field Type** | **Zise** | **Description** |
| Id (pk) | Int | N/A | Unique identifier for specific notification |
| User\_id (fk) | Int | N/A | Unique identifier for the user |
| Title | Varchar | 50 | Specific title |
| Message | Text | 255 | Specific message for notification |
| Link | Varchar | 255 | Specific link for notification |
| Is\_read | Boolean | N/A | To identify if it’s done to read |
| Created\_at | Date/time | N/A | Date/time receive notification |

**Table 5. Data Dictionary of TBL\_Notification**

**Output and User Interface**

This is the process of designing user interface of the project. It serves as a blueprint on how the project’s user interface will look like. Below are the figures of showing.

**Admin**

Email

Password

**Figure 15. Admin Login Page**

Shows the login page for DPWH admins. It's a simple screen with just two boxes - one for email and one for password. This page is only for DPWH staff who need to manage road problems in the system

**DASHBOARDS**

Road Sense

Admin Account

Dashboard

Hazard Management

Feedback Reports

Reports

Logout

Admin:

**Figure 16. Admin Dashboard Page**

Shows the part of the system for DPWH road workers. This big empty space on the right side is where DPWH people will see pictures and numbers about road holes. When it's working, this spot will show easy-to-understand charts about where the bad roads are. DPWH folks can quickly see stuff like how many holes were found today, which streets have the most problems, and which reports are brand new. The dashboard shows all the important road stuff in one place so DPWH doesn't have to read lots of papers. This helps them decide super-fast which roads to fix first so they can send their fix-it trucks to the worst spots right away.

**HAZARD MANAGEMENT**

Road Sense

Dashboard

Hazard Management

Feedback Reports

Reports

Logout

Admin:

**Figure 17. Admin Hazard Management Page**

Show the page for DPWH road workers. This page is mostly for checking if road problems are real and updating them when they're fixed. When DPWH staff visit a road hole that someone reported, they come back to this page to say if it's really there. If they find the hole is real, they mark it as "verified" so an admin can approve it. Then when DPWH sends workers to fix the hole, they come back to this same page to update the status to "resolved" or "fixed." This helps everyone know which problems are real and which ones are already taken care of.

**REPORTS**

Road Sense

Dashboard

Hazzard Management

Feedback Reports

Reports

Logout

Admin:

**Figure 18. Admin Reports page**

Shows the page where DPWH road workers can see all road problems that were ever reported. This page is like a big picture book of all the holes and bad spots on roads. DPWH can see a list of every report that's been made. DPWH can pick any dates they want (like January to March) and print out a paper showing all the problems from that time. They can make papers for their bosses showing how many holes they fixed last month. The reports page helps DPWH show they're doing good work and helps them know which roads need more fixing. They can see which roads break a lot and might need to be totally rebuilt. This page is like looking at the whole history of road problems so DPWH can fix roads better.

**FEEDBACK REPORTS**

Road Sense

Dashboard

Hazard Management

Feedback Reports

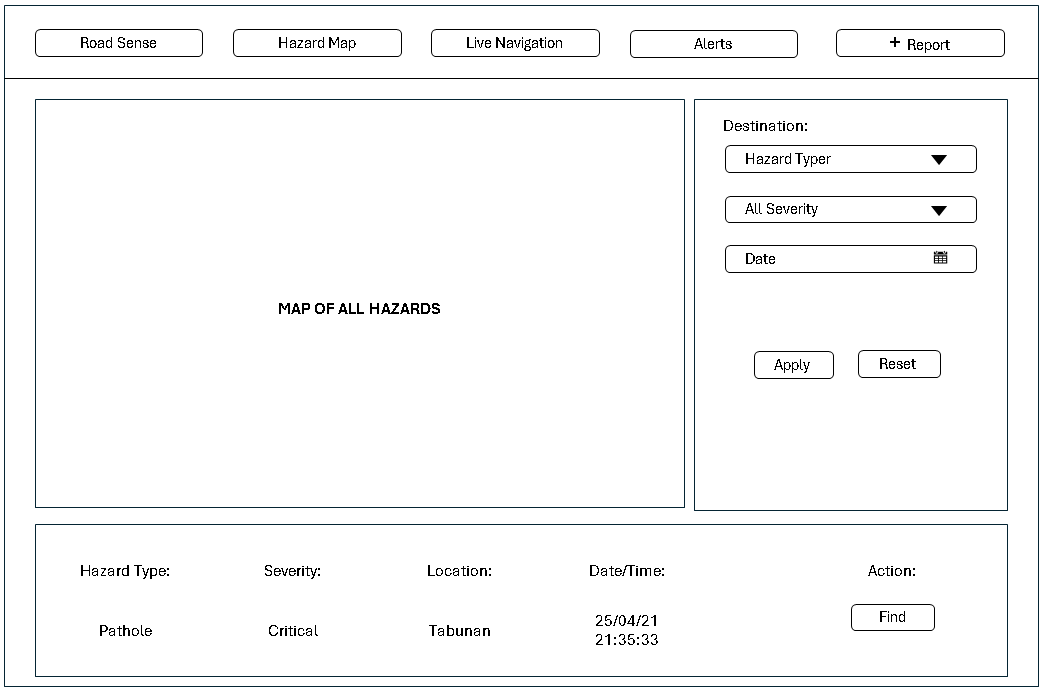
Reports

Logout

**Admin:**

**Figure 19. Admin Feedback Report Page**

Shows the page where DPWH road workers can see what people say about road fixes. This page shows when fixed holes come back again. When DPWH fixes a hole, sometimes the same hole breaks open again after rain or big trucks. On this page, drivers can tell DPWH "That hole you fixed is back!" DPWH can read these notes and learn which fixes didn't work well. They can see stuff like "holes on Main Street always come back" or "fixes done when it's rainy don't stay fixed." This helps DPWH know when they need to do bigger, better fixes instead of quick patches. The feedback page helps DPWH stop fixing the same holes over and over, and instead fix roads in ways that last longer.

****

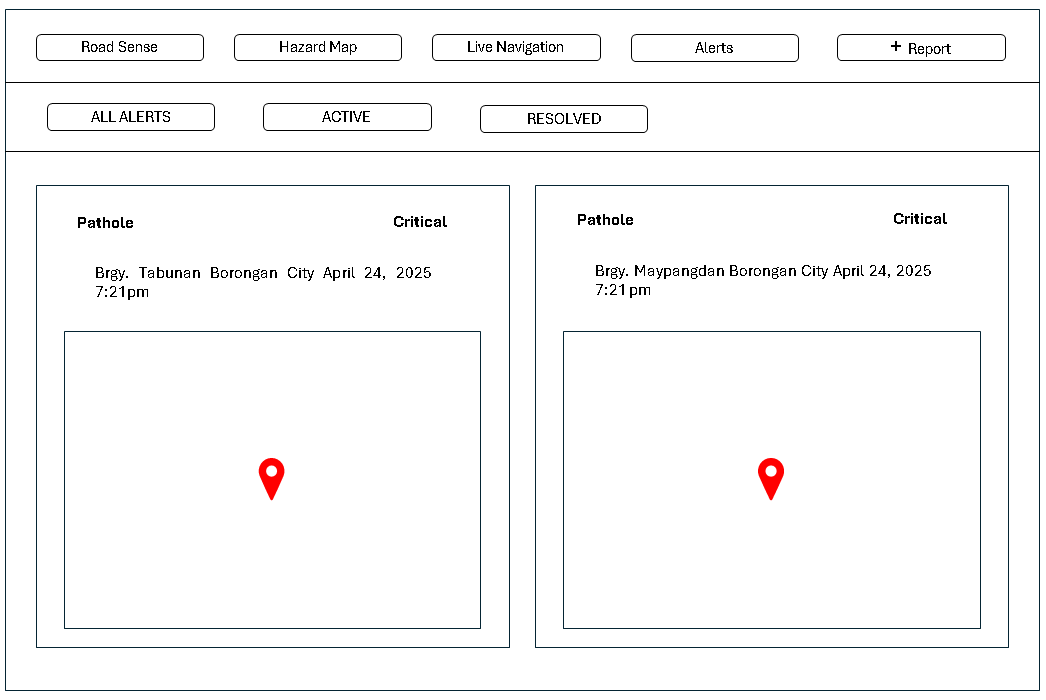
**Figure 20. User Map of all Hazard Page**

Hazard map shows the page where regular people see all road problems in the Road Sense system. At the top are buttons for different features like the hazard map, live directions, alerts, and reporting new problems. The middle has a big area labeled "MAP OF ALL HAZARDS" that will show dots for each road problem. On the right side, users can filter what they want to see by picking hazard types (like potholes or floods), how bad the problems are, and when they were reported. At the bottom is an example of a critical pothole in Tabunan reported on April 25, 2021, with a "Find" button to locate it on the map. This page helps drivers see where bad roads are so they can plan safer trips and avoid damaging their cars or having accidents.

****

**Figure 21. User Live Navigation Page**

Shows the Live page where drivers can see their trip on a map. The red pin shows exactly where you are right now, and the system updates this pin as you move. The blue line shows the safe path to follow to get to the green pin (where you want to go). The cool thing about this page is it tracks where you are all the time, like a GPS. When you're driving and get close to a pothole or bad road spot (about 100 meters away), the system will talk to you with a voice alert saying stuff like "Warning: Large pothole ahead" or "Caution: Flooded area in 100 meters." you can just listen to the warnings and slow down or change lanes. The voice alerts help keep your eyes on the road while still avoiding the bad spots that could damage your car.

****

**Figure 22. User Alert Page**

Shows the page that helps you stay updated on all the problems happening now and the ones that are already fixed. It shows stuff like potholes, road cracks, and other issues, and each one has a box that tells you exactly where it is. You can see if something is still a problem or if it's already fixed, and you can check where it is on the map. This way, you know right away where the trouble spots are, helping you stay safe and plan your route. The page is really useful for keeping you informed and helping you avoid any hassles on the road.

**Data Flow Diagram**

Data flow diagram shows the way information flows through a process or system. It includes data inputs and outputs, data store, and the various sub processes the data moves through. DFDs are built using standardized symbols and nation to describe various entities and their relationships.

**Figure Definition**

These are the common symbols that were use to present the specific that the actors had taken in using the entire project. This was presented through a Data Flow Diagram.

**Symbols**

**Description**

Source (Sink)

Process

The source of System inputs or sink of system outputs

Step-by-step instructions are followed that transform inputs (a computer or a person or t=both doing the work)

Data Store

An open-ended rectangle represents the location where data is stored

Data Flow Symbol

Represent a pathway for data

0

Road Pathole and Traffic Hazard Mapping

ADMIN

USERS

ALERTS

HAZARD MAP

RECEIVE INCEDENT REPORT

SUBMIT INCEDENT REPORT

LIVE HAZARD MAP NAVIGATION

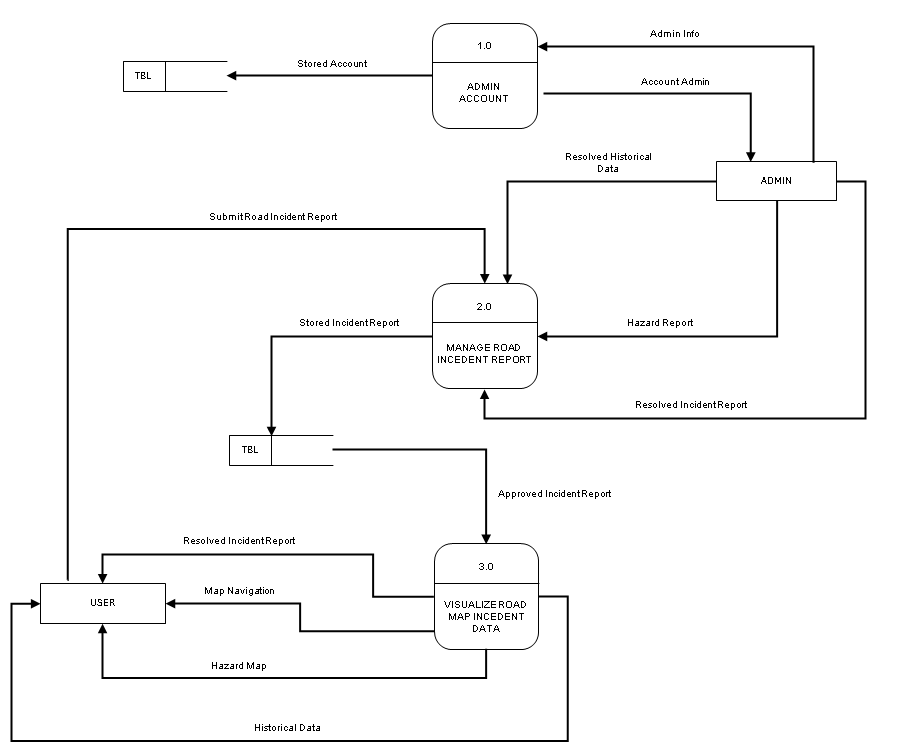
RESOLVE INCEDENT

HISTRICAL DATA

VERIFIED HAZARD REPORT

**Figure 23 Level 0 of Context Diagram**

Discusses the live hazard map navigation system, showing how users interact with the hazard map to receive real-time alerts and navigate safely. Users can submit incident reports, which are processed to update the hazard map with road pathole and traffic hazard data. The system receives these incident reports, and admins play a crucial role by verifying the reports to ensure accuracy. Admins also resolve incidents and maintain historical data for analysis, contributing to a reliable and up-to-date hazard mapping system that aids in efficient emergency response and navigation.

****

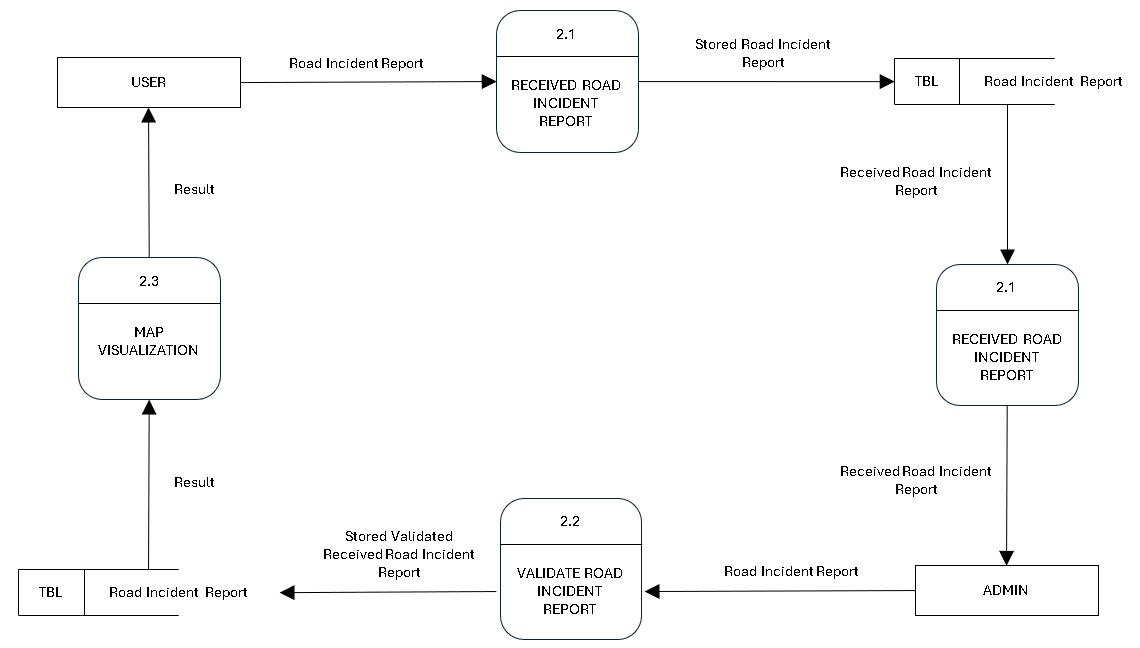
**Figure 24. Data Flow Diagram Level-0 Context Diagram**

Shows first, the Admin Account (1.0) keeps track of admin users and saves their info so they can log in and run things. Second, the Manage Road Incident Report (2.0) is where people talk about holes or floods they see on roads, and these reports get saved in a database where admins can check if they're real problems. Third, the Visualize Road Map (3.0) turns all these checked reports into maps that anyone can use to avoid bad roads. This system is super important because it helps the DPWH (Department of Public Works and Highways) quickly find which roads need fixing most. The DPWH can see where lots of people report problems, so they know exactly where to send repair teams first. This saves government money by fixing the worst spots first and helps them plan better for future road work. It also shows the DPWH which roads break often and might need full rebuilding instead of just patches.

****

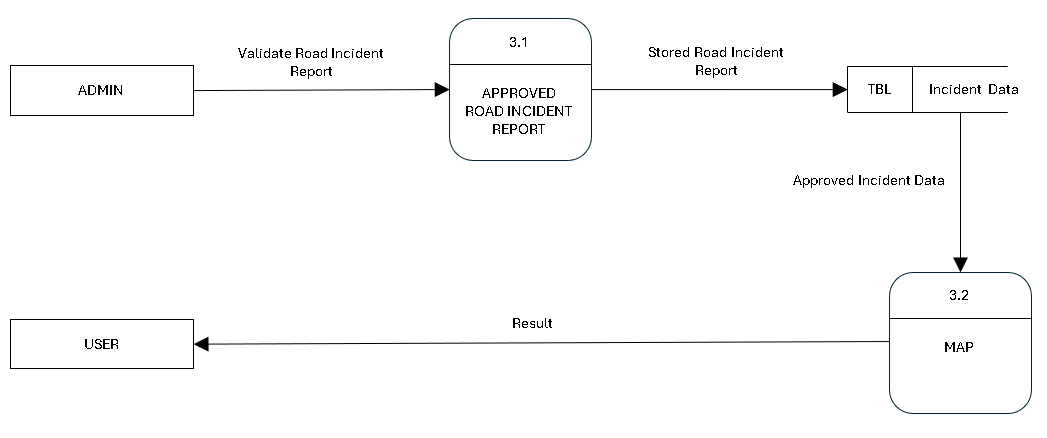
**Figure 25. Level 1 Data Flow Diagram**

Shows how admins log in to system. First, the admin types their username and password, which goes to the Accept Data (1.1) part. Then, this information is updated and sent to the Validate Data (1.2) part, which checks if the login information is correct. If it's right, the system saves this validated data in a table (TBL). The Accept Data part then gets this stored information from the table and sends back a result to the admin, either letting them in or keeping them out. This login process is important because it makes sure only real DPWH workers or city officials can approve road repairs, update or verify the hazard reports, or see all the data about where roads need fixing. Without this security step, anyone could change the system and road repair teams might go to the wrong places.

****

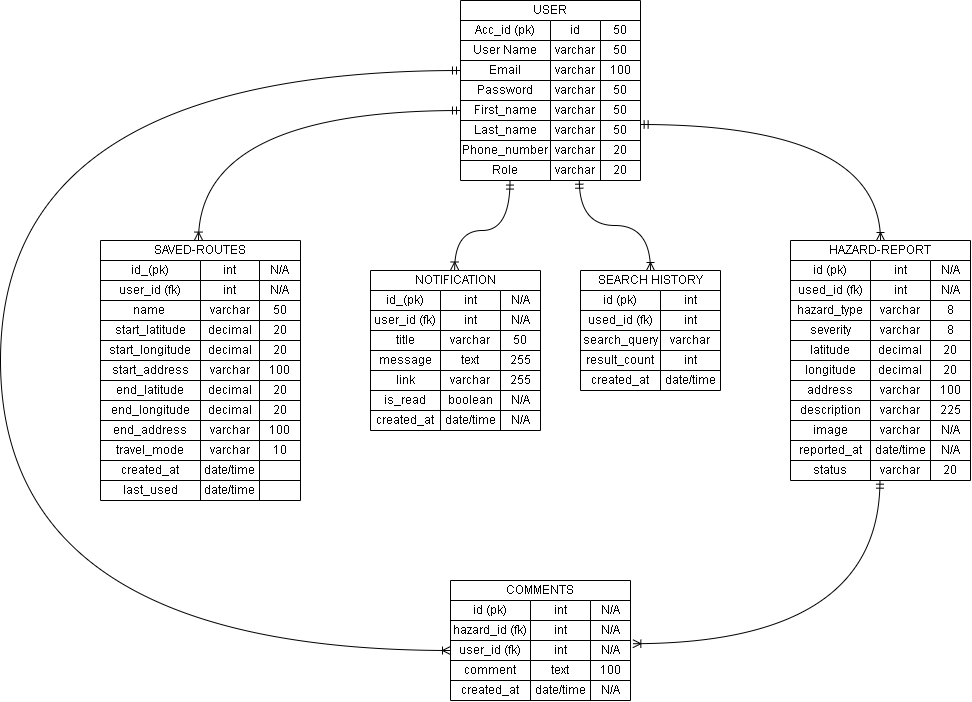
**Figure 26. Level 2 Data Flow Diagram**

Shows how road pathole or hazards reports work in the system. First, a normal person tells the system about a bad road. The system saves this report (2.1) in a big list. Then the saved report goes to the checking part (2.2) where someone makes sure it's real. Next, an admin person looks at it and sends the real reports to the approval part (2.3). After approval, the report gets saved again as a for-sure road problem. Then the road problem shows up on maps (2.4) that everyone can see. This whole thing is super helpful for DPWH (road fixing people) because they don't waste time on fake reports. They can see which roads are really bad and need fixing first.

****

**Figure 27. Level 3 Data Flow Diagram**

Shows the last part of how road problems get on maps. First, the admin person checks road incident reports and sends good ones to the Approved Road Incident Report part (3.1). These approved reports get saved in a big list (TBL) as official incident data. Then the approved data goes to the Map part (3.2), which turns all these road problems into a map that's easy to understand. The map then shows the result to regular users so they can see where the bad roads are. This is really helpful for DPWH because it turns all their work checking reports into something that helps drivers right away. The map shows exactly where the problems are so DPWH repair trucks can find them easily.

****

**Figure 28. Entity Relationship Diagram**

**Programming Environment**

**HTML (HyperText Markup Language)** - HTML makes the basic structure of our Road Sense website. It creates all the parts you see like the login boxes, buttons, maps, and report forms. We use HTML to make sure everything is in the right place and easy to find. It's like the skeleton that holds everything together and lets us add the other cool stuff like colors and maps.

**CSS (Cascading Style Sheets)** - CSS makes our Road Sense system look nice. It adds colors, changes the size of things, and makes buttons look clickable. We use CSS to make sure our website looks good on phones and computers. We use Tailwind CSS framework to quickly build good-looking pages without writing lots of custom CSS. Tailwind lets us add styles directly in our HTML with simple class names, which helps us build the interface faster.

**JavaScript** - JavaScript makes our Road Sense maps work and do cool things. It helps show where potholes are on the map, updates your location as you drive, and makes the voice alerts talk to you when you get near a road problem. JavaScript also makes it so you can click on things and they change without loading a new page. It makes the website feel more like a real app.

**PHP (Hypertext Preprocessor)** - PHP works behind the scenes in Road Sense. When you click "login" or "report pothole," PHP checks if your information is right and saves new reports to our database. It's the part that talks between what you see on screen and where we store all the information about roads and users. PHP helps make sure only real DPWH workers can approve reports.

**MySQL** - MySQL is where all Road Sense information gets saved. It stores user accounts, all reported potholes, which ones are fixed, and where everything is located. MySQL keeps track of all this info in organized tables so we can quickly find what we need. It's like a super organized filing cabinet for all our road problem data.

**Leaflet Map**- Leaflet helps us show maps in Road Sense. It creates the interactive maps where users can see reported problems and their current location. Leaflet makes it easy to add markers for potholes, show different colors for how bad problems are, and draw basic map features. It works well on phones and computers, which is important since many people will use Road Sense while traveling.

**TomTom API** - TomTom API gives our Road Sense system super smart navigation features. It figures out the best routes to avoid reported road problems, calculates how long trips will take, and shows traffic information. TomTom helps give users turn-by-turn directions and powers the voice alerts that warn drivers about road hazards ahead. It works with our Leaflet maps to give users a complete picture of road conditions and safe travel options.

**Evaluation**

The respondents evaluated the system using the following scale:

|  |  |
| --- | --- |
| **Level of Agreement** | **Scale Description** |
| 5 | Strongly Agree |
| 4 | Agree |
| 3 | Fair |
| 2 | Disagree |
| 1 | Strongly Disagree |

**System Testing (Alpha Testing)**

The obtained mean was interpreted using the following:

**Functional Suitability**

|  |  |
| --- | --- |
| **Numerical Equivalent** | **Interpretation** |
| 4.20 – 5.00 | Highly Functional |
| 3.40 – 4.19 | Functional |
| 2.60 – 3.39 | Moderately Functional |
| 1.80 – 2.59 | Slightly Functional |
| 1.00 – 1.79 | Poorly Functional |

**Performance Efficiency**

|  |  |
| --- | --- |
| **Numerical Equivalent** | **Interpretation** |
| 4.20 – 5.00 | Highly Efficient |
| 3.40 – 4.19 | Efficient |
| 2.60 – 3.39 | Moderately Efficient |
| 1.80 – 2.59 | Slightly Efficient |
| 1.00 – 1.79 | Inefficient |

**Compatibility**

|  |  |
| --- | --- |
| **Numerical Equivalent** | **Interpretation** |
| 4.20 – 5.00 | Highly Compatible |
| 3.40 – 4.19 | Compatible |
| 2.60 – 3.39 | Moderately Compatible |
| 1.80 – 2.59 | Slightly Compatible |
| 1.00 – 1.79 | Incompatible |

**Reliability**

|  |  |
| --- | --- |
| **Numerical Equivalent** | **Interpretation** |
| 4.20 – 5.00 | Highly Reliable |
| 3.40 – 4.19 | Reliable |
| 2.60 – 3.39 | Moderately Reliable |
| 1.80 – 2.59 | Slightly Reliable |
| 1.00 – 1.79 | Unreliable |

**Security**

|  |  |
| --- | --- |
| **Numerical Equivalent** | **Interpretation** |
| 4.20 – 5.00 | Highly Secured |
| 3.40 – 4.19 | Secured |
| 2.60 – 3.39 | Moderately Secured |
| 1.80 – 2.59 | Slightly Secured |
| 1.00 – 1.79 | Unsecured |

**Acceptance testing (Beta testing)**

The obtained mean was interpreted using the following:

|  |  |
| --- | --- |
| **Numerical Equivalent** | **Interpretation** |
| 4.20 – 5.00 | Highly Acceptable |
| 3.40 – 4.19 | Acceptable |
| 2.60 – 3.39 | Moderately Acceptable |
| 1.80 – 2.59 | Slightly Acceptable |
| 1.00 – 1.79 | Unacceptable |

**Research Locale GIS map**

**Research Design**

The study used the developmental-evaluation research design approach as the system was subjected to end user evaluation.

**Research Locale**

The study was conducted at the Department of Public Works and Highways Eastern Samar District Engineering Office Borongan City Eastern Samar, Region VIII.

|  |  |
| --- | --- |
|  | |
| **RESEARCH LOCALE** | **Republic of the Philippines Easter Visayas**  **Region 8** |
| Marker with solid fill**Legend:**  Department of Public Works and Highways Eastern Samar District Engineering Office Borongan City Eastern Samar, Region VIII.  Borongan City Google Satellite. |
| **Data Source:**  QGIS Desktop 3.40.1 Google Satellite  Cartographer: Glorioso Tabay Jr |

**Table 1. Research Locale**

**Research Respondents**

We will pick 15 people to test our road hole map website. First, we will get 5 people from DPWH (the road fixing office) in Eastern Samar who know all about fixing roads. Second, we will get 5 instructors from the College of Computer Studies (CCS) at Eastern Samar State University who are professionals in web development and can give expert suggestions to improve our proposed website. Third, we will get 5 regular people who drive or walk on roads every day and can report potholes and hazards they find. We will need all these different people because they will help in different ways. The DPWH folks will tell us if our road hole information is right. The computer professionals from ESSU will give valuable technical advice on our system design and features. The regular road users will test how easy it is to report problems and use the map while traveling. Getting help from all these different people will make our road hole map better for everyone. They will tell us what needs fixing so the website really helps people avoid bad spots on roads.

|  |  |
| --- | --- |
| **RESPONDENTS** | **NUMBERS OF REPONDENTS** |
| DPWH Maintenance Engineers | 5 |
| Faculty Members from CCS | 5 |
| Regular Road Users (Drivers/Pedestrians) | 5 |
| **TOTAL** | 15 |

**Table 2. Distribution of Respondents**

**Instrumentation**

In this phase, the system was evaluated using two instruments: the ISO/IEC 25010:2011 Software Evaluation Questionnaire for quality metrics during system testing and the System Usability Scale (SUS) for acceptance testing.

The ISO questionnaire includes a series of questions designed to assess user satisfaction across various dimensions, such as functional usability, performance efficiency, compatibility, usability, and reliability. This helps evaluate the system’s overall quality during the testing phase.

The System Usability Scale (SUS) consists of questions that focus on user experience, addressing aspects like system complexity, usage frequency, consistency, and overall satisfaction. By using these standardized evaluation frameworks, the study aims to gather valuable insights into the system’s performance, ensuring it meets the desired quality standards and effectively addresses the diverse needs and expectations of its users.

**Data Analysis**

Statistics using the mean was employed to present the demographic characteristics of the respondents and the level of the system effectiveness. The data from the ISO/IEC 25010:2011 Software Evaluation Questionnaire and System Usability Scale (SUS) will be analyzed to assess the system’s quality and usability during the alpha test, and the selected respondents during the beta test.

**Mean**

This is the average of the score – the mathematical center of a distribution. It used symmetrical, unimodal distributions of interval or ration scores.

The ISO's formula mean is:

X = Σ *x/n* Σx = sum of all scores

*n*= number of scores

Where:

Q1-10: Scores for each question

The IBM formula mean is:

SUS = (Q1 adj + Q2 adj + Q3 adj + ... + Q10 adj) x (2.5)

Where:

Q1-10: Scores for each