AMRnet

The AMRnet Team

May 19, 2026

DASHBOARD USERS

1 Citation, Code and License **3**

2 Disclaimer **5**

2.1 Dashboard overview 5
2.2 Individual pathogen details 7
2.3 Source data & tools 15
2.4 Data access 17
2.5 Developer Guide 20

The AMRnet dashboard aims to make high-quality, robust and reliable genome-derived AMR surveillance data accessible to a wide audience. Visualizations are geared towards showing national annual AMR prevalence estimates and trends, that can be broken down and explored in terms of underlying lineages/genotypes and resistance mechanisms. We do not generate sequence data, but we hope that by making publicly deposited data more accessible and useful, we can encourage and motivate more sequencing and data sharing.

We started with *Salmonella* Typhi, based on our TyphiNET dashboard which uses data curated by the Global Typhoid Genomics Consortium (to improve data quality and identify which datasets are suitable for inclusion) and analysed in Pathogenwatch (to call AMR determinants and lineages from sequence data). Dashboards are now available for *Klebsiella pneumoniae*, *Neisseria gonorrhoeae*, non-typhoidal *Salmonella*, *E. coli* and *Shigella* using data sourced from the Pathogenwatch and Enterobase platforms. In the future, we plan to add additional organisms, sourced from these and other platforms.

A major barrier to using public data for surveillance is the need for careful data curation, to identify which datasets are relevant for inclusion in pooled estimates of AMR and genotype prevalence. This kind of curation can benefit a wide range of users and we plan to work with other organism communities to curate data, and to contribute to wider efforts around metadata standards. Please get in touch if you would like to work with us (amrnetdashboard@gmail.com).

Key links:

- The dashboard is accessible at https://www.amrnet.org.
- Info about the project team (based at London School of Hygiene and Tropical Medicine), and our policy advisory group, is here.

CHAPTER

ONE

CITATION, CODE AND LICENSE

AMRnet is free and open source software, distributed under the terms of the *GPLv3 License*.

- The dashboard code is open access and available in GitHub.
- Issues and feature requests can be posted here.
- API access is described on the *Data access* page.
- If you want to install the AMRnet code to develop your own dashboard instances, see the *Installation Guide*.

If you use the AMRnet website or code, please cite **AMRnet** Cerdeira LT, Dyson ZA, Sharma V, et al. AMRnet: a data visualization platform to interactively explore pathogen variants and antimicrobial resistance. Nucleic Acids Res. Published online November 6, 2025.doi:[10.1093/nar/gkaf1101]. Depending on what data and visualizations you are using in AMRnet, you should also consider citing the upstream source databases and typing tools (noted in the pathogen-specific pages and *here*), or individual source studies (via PubMed IDs or DOIs available in the TSV download at the bottom of each dashboard).

CHAPTER

TWO

DISCLAIMER

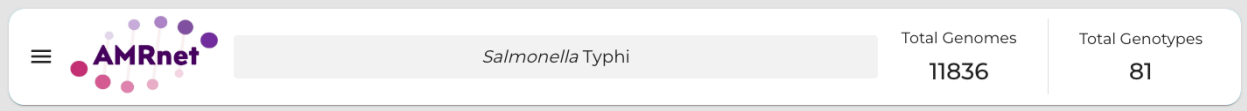
AMRnet is an open source research project, providing access to data and visualisations sourced from publicly accessible sequence databases and analysis platforms. The content on our service is provided for general information only. It is not intended to amount to advice on which you should rely. You must obtain professional or specialist advice before taking, or refraining from, any action on the basis of the content on our service. Although we make reasonable efforts to update the information on our service, we make no representations, warranties or guarantees, whether express or implied, that the content on our service is accurate, complete or up to date. We are not responsible for the results of reliance on any such information.

Refer to the *License* for clarification of the conditions of use.

2.1 Dashboard overview

2.1.1 Header

Use the menu to **select a species or pathogen group** to display. Each pathogen has its own dashboard configuration that is customised to show genotypes, resistances and other relevant parameters. Numbers indicate the total number of genomes and genotypes currently available in the selected dashboard.



2.1.2 Map

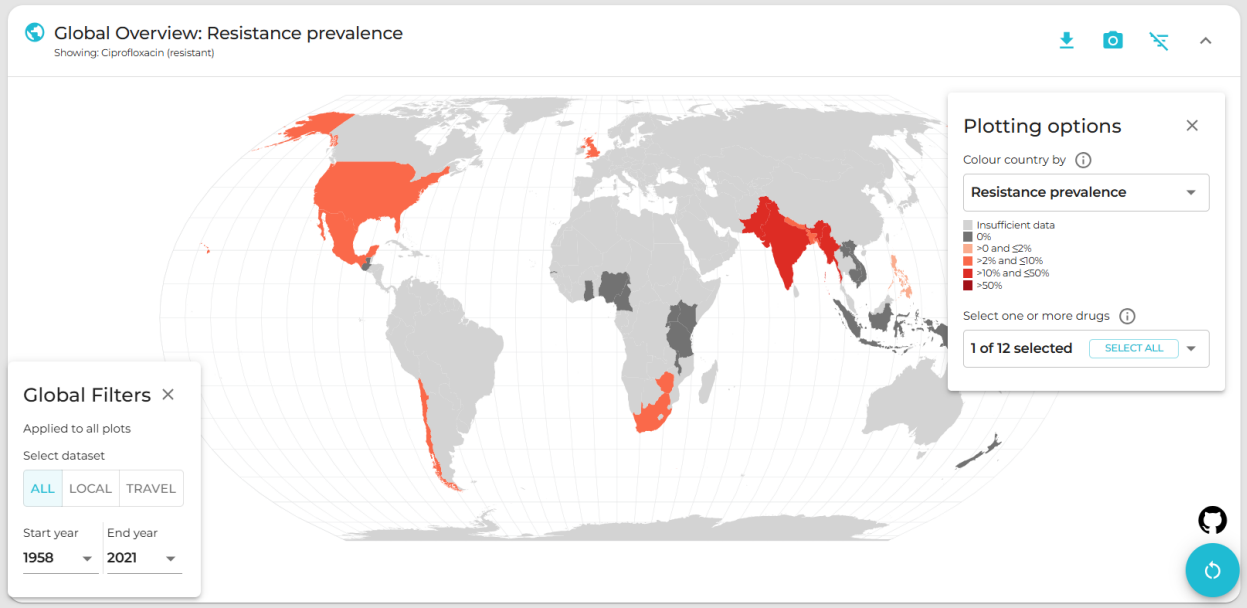
Use the **Plotting options** panel on the right to **select a variable to display per-country summary data** on the world map. Prevalence data are pooled weighted estimates of proportion for the selected resistance or genotype. A country must have N20 samples (using the current filters) for summary data to be displayed otherwise, it will be coloured grey to indicate insufficient data are available.

Use the **Global Filter** panel on the left to recalculate summary data for a specific time period and/or subgroup/s (options available vary by pathogen). Filters set in the **Global Filter** panel apply not only to the map, but to all plots on the page.

Clicking on a country in the map also functions as a filter, so that the **Summary plots** in the panels below reflect data for the selected country only. Per-country values displayed in the map can be downloaded by clicking the downward-arrow button at the top right of the panel.

Plot-specific downloads: The summary statistics currently displayed in the plot (e.g. resistance prevalence per country; with numerator, denominator and percentage) can be downloaded in tabular (TSV) format by clicking the down arrow

(top-right of panel). A static image (PNG format) of the current map view can be downloaded by clicking the camera icon.



2.1.3 Summary plots

This panel offers a series of summary plots. The default view is "AMR trends". **Click a plot title** in the rotating selector at the top of the panel, to choose a different plot. The specific plots available vary by pathogen, as do the definitions of AMR and genotype variables (see individual *pathogen details*). Summary plots are intended to show region- or country-level summaries, but if no country is selected they will populate with pooled estimates of proportion across all data passing the current filters.

All plots are interactive. Use the **Plotting options** panel on the right to modify the region/country to display, or to select other options available for the current plot such as which variables to display, and whether to show **counts or percentages**.

Line and bar plot have a dynamic legend to the right; click on an x-axis value to display counts and percentages of secondary variables calculated amongst genomes matching that x-axis value. For example, all pathogens have an 'AMR by genotype' plot; click a genotype to display counts and percentages of resistance estimated for each drug or class.

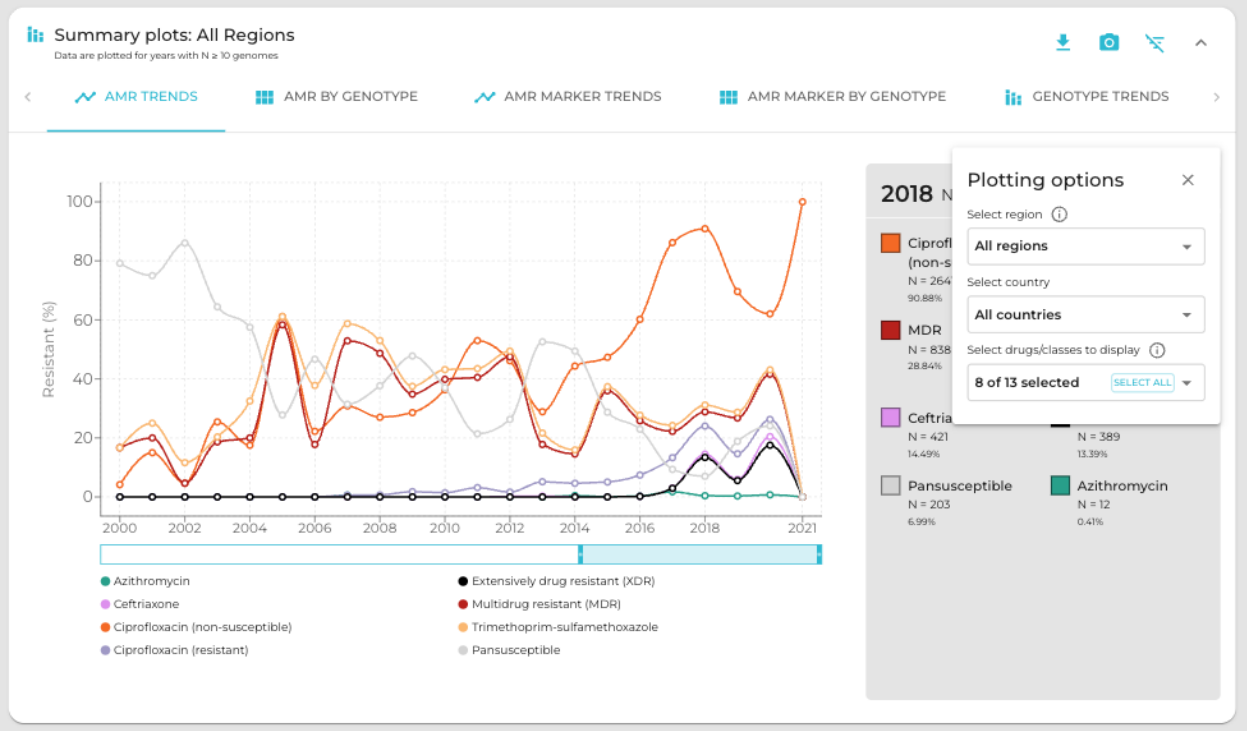
Plot-specific downloads: Summarised values displayed in the current plot can be downloaded by clicking the down arrow button (top-right of panel). A static image (PNG format) of the current plotting view can be downloaded by clicking the camera icon.

2.1.4 Geographic comparisons

This panel uses heatmaps to explore how the prevalence of AMR or genotypes varies between countries or regions. The default view is AMR prevalence per country.

Use the **Plotting options** panel on the right to stratify rows by region rather than country, to view genotype prevalence instead of AMR (in columns). For some organisms, there is also the option to plot columns as resistance markers (for a selected drug).

Plotting options can also be modified to restrict the view to selected countries or regions (in rows), or to specific subsets of genotypes or drugs (in columns). Countries are not plotted (and not available to select) if the number of samples passing the current global filter is below 20. By default, the 20 most common genotypes or drugs (passing



current filters) will be included in the plot; in the drop-down list to modify the specific set of drugs or genotypes shown, the available options will be ordered by their frequency in the current selection.

Plot-specific downloads: Summarised values displayed in the current heatmap can be downloaded by clicking the down arrow button (top-right of panel). A static image (PNG format) of the current plotting view can be downloaded by clicking the camera icon.

2.1.5 Downloads

At the bottom are buttons to download (1) the individual genome-level information that is used to populate the dashboard ('Download database (TSV format)'); and (2) a static report of the currently displayed plots, together with a basic description of the data sources and variable definitions ('Report view').

2.2 Individual pathogen details

Each dashboard is populated a bit differently, from different sources and with different inclusion criteria, using AMR genotyping and lineage definitions specific to the pathogen.

Select a pathogen below to see the details:

- *Escherichia coli*
- *Escherichia coli* (diarrheagenic)
- *Shigella* + EIEC
- *Klebsiella pneumoniae*
- *Neisseria gonorrhoeae*
- *Salmonella* (non-typhoidal)
- *Salmonella* (invasive non-typhoidal)


Geographic Comparisons
Data are restricted to the Global filters selected (Year 1958 - 2021) , and regions/countries with N≥20 passing these filters.
HEATMAP
CipNS CipR CRO AMP/AMX AZM CHL SUL SXT TET TMP
Eastern Africa 19.27 0.06 0.06 80.54 0 79.13 80.22 80.15 38.28 80.73
Latin America and the Caribbean 15.1 2.35 0.34 1.01 0 1.01 1.01 0.67 0.67 0.67
Melanesia 0.83 0 0 1.24 0 1.38 1.52 1.38 0 1.38
Northern America 29.29 4.29 0.71 8.57 0 7.86 10.71 10 2.14 10
Northern Europe 61.06 7.96 0.88 22.12 0 20.35 22.12 22.12 4.42 23.01 20.35 0 27.43
Polynesia 4.55 0 0 0 0.32 0 0 0 0 0 0 0 95.45
South-eastern Asia 35.04 1.83 0.1 36.97 0.19 35.81 36.39 35.62 35.71 35.81 34.65 0 47.2
Southern Africa 15.34 3.19 0 59.42 0 56.87 60.06 59.11 11.5 59.11 56.23 0 37.38
Southern Asia 76.4 19.28 8.51 24.62 0.79 24.56 26.52 24.64 2.05 24.67 22.74 7.93 4.16
Western Africa 10.48 0 0 50.48 0 47.62 57.46 57.14 50.16 57.46 40.32 0 40.63
0% 1% 100%
Plotting options
Select region/country
Region
Select countries/regions
10 of 14 selected
SELECT ALL
Columns
Resistance prevalence
Select drugs
13 of 13 selected
CLEAR ALL
Download database (TSV format)
Download PDF
Info and Definitions

- *Salmonella* Typhi

A list of all upstream databases and tools, with links and citations, is available on the *Source data & tools* page.

2.2.1 AMR definitions based on Enterobase/AMRfinderplus data

These definitions are used in the *E. coli*, *Shigella*, and non-typhoidal *Salmonella* dashboards, where data are populated from Enterobase which uses AMRfinderplus to identify AMR determinants from genome assemblies. For other organisms, populated from Pathogenwatch, see the pathogen-specific pages above for details of how AMR is detected/defined.

Resistance indicator variable	Definition	Enterobase column/s	AMRFinderPlus Class/Subclass
Ampicillin	1 marker from any column	Penicillin, Carbapenemase, ESBL	Subclass: BETA-LACTAM
Aminoglycosides	1 marker	Aminoglycoside	Class: AMINOGLYCOSIDE
Carbapenems	1 marker	Carbapenemase	Subclass: CARBAPENEM
Ciprofloxacin (non-susceptible)	1 marker	Quinolone	Class: QUINOLONE
Chloramphenicol	1 marker	Phenicol	Class: PHENICOL
Colistin	1 marker	Colistin	Class: COLISTIN
ESBL	1 marker from any column	ESBL, Carbapenemase	Subclass: CEPHALOSPORIN
Fosfomycin	1 marker	Fosfomycin	Class: FOSFOMYCIN
Macrolide	mph(A) or acrB_R717	Macrolide	Class: MACROLIDE
Tetracycline	1 marker	Tetracycline	Class: TETRACYCLINE
Tigecycline	1 marker	Other Classes: Tigecycline	Subclass: TIGECYCLINE
Trimethoprim	1 marker	Trimethoprim	Class: TRIMETHOPRIM
Trimethoprim-Sulfamethoxazole	1 marker from each column	Trimethoprim, Sulfonamide	Class: TRIMETHOPRIM, SULFONAMIDE
Pansusceptible	0 markers	all AMR columns	—

Escherichia coli

Escherichia coli data in AMRnet are drawn from Enterobase, which calls AMR genotypes using NCBI's AMRFinderPlus and assigns lineages using MLST, cgMLST and hierarchical clustering. Last update: 5 August 2025.

> **Warning**
>
> The *E. coli* data used in AMRnet are not yet curated for purpose-of-sampling, and therefore reflect the biases of global sequencing efforts which may be skewed towards sequencing AMR strains and/or outbreaks. Data curation efforts are ongoing however until then, please be careful when interpreting the data in the dashboard.

Variable definitions

- **Lineages**: Lineages are labeled by 7-locus sequence type (ST).
- **AMR determinants**: Enterobase identifies AMR determinants using NCBI's AMRFinderPlus run with default settings. AMRnet imports these AMR genotype calls, and assigns them to drugs/classes in the dashboard using the Subclass curated in refgenes. see *the table*

Escherichia coli diarrheagenic

Enterobase classifies *E. coli* genomes to pathotypes using this logic. Pathotypes included in the *E. coli* (diarrheagenic) dashboard are:

- Shiga toxin-producing *E. coli* (STEC)
- Enterohemorrhagic *E. coli* (EHEC)
- Enterotoxigenic *E. coli* (ETEC)
- Enteropathogenic *E. coli* (EPEC)
- Enteroinvasive *E. coli* (EIEC)

Last update: 5 August 2025.

Warning

The *E. coli* data used in AMRnet are not yet curated for purpose-of-sampling, and therefore reflect the biases of global sequencing efforts which may be skewed towards sequencing AMR strains and/or outbreaks. Data curation efforts are ongoing however until then, please be careful when interpreting the data in the dashboard.

Variable definitions

- **Lineages**: Lineages are labeled by the pathovar followed by the (7-locus) ST.
- **AMR determinants**: Enterobase identifies AMR determinants using NCBI's AMRFinderPlus run with default settings. AMRnet imports these AMR genotype calls, and assigns them to drugs/classes in the dashboard using the Subclass curated in refgenes, see *the table*.

Shigella + EIEC

Shigella and enteroinvasive *E. coli* (EIEC) data in AMRnet are drawn from Enterobase, which calls AMR genotypes using NCBI's AMRFinderPlus and assigns lineages using cgMLST and hierarchical clustering. Last update: 5 August 2025.

Warning

The *Shigella* + EIEC data used in AMRnet are not yet curated for purpose-of-sampling, and therefore reflect the biases of global sequencing efforts which may be skewed towards sequencing AMR strains and/or outbreaks. Data curation efforts are ongoing however until then, please be careful when interpreting the data in the dashboard.

Variable definitions

- **Lineages**: The logic used by Enterobase to classify genomes as *Shigella* or EIEC are detailed here. *Shigella sonnei* are monophyletic and labelled as lineage 'S. *sonnei*'. For other *Shigella*, lineages are labeled by the species followed by the HC400 (HierCC) cluster ID (as this nomenclature has been shown to mirror the paraphyletic lineage structure of *Shigella*). EIEC lineages are labeled by ST (e.g. 'EIEC ST99').
- **AMR determinants**: Enterobase identifies AMR determinants using NCBI's AMRFinderPlus run with default settings. AMRnet imports these AMR genotype calls, and assigns them to drugs/classes in the dashboard using the Subclass curated in refgenes, see *the table*.

Klebsiella pneumoniae

Klebsiella pneumoniae data are sourced from Pathogenwatch, which calls AMR (using Kleborate) and genotypes (MLST) from genomes assembled from public data. Last update: 5 August 2025.

> **Warning**
>
> The *Klebsiella pneumoniae* data used in AMRnet are not yet curated for purpose-of-sampling, and therefore reflect the biases of global sequencing efforts which have been largely directed at sequencing ESBL and carbapenemase-producing strains or hypervirulent strains. Data curation efforts are ongoing however until then, please be careful when interpreting the data in the dashboard.

Variable definitions

- **Genotypes**
  - ST: sequence type, defined by the 7-locus MLST scheme for *Klebsiella pneumoniae* species complex
  - cgST: core genome sequence type, defined by the core genome MLST scheme, maintained by Institut Pasteur
  - Sublineage: defined from core-genome MLST, as described here
- **AMR determinants** are called using Kleborate v3, described here. Ciprofloxacin resistance is defined based on the prediction generated by Kleborate. For all other drugs, resistance is defined by the presence of one or more markers in the corresponding Kleborate output column.
- **Pansusceptible**: no resistance determinants identified besides a wildtype beta-lactamase SHV allele associated with intrinsic resistance to ampicillin (i.e. not an ESBL or inhibitor-resistant variant of SHV, see Tsang et al 2024.

Abbreviations

- **ESBL**: extended-spectrum beta-lactamase
- **ST**: sequence type
- **cgST**: core-genome sequence type

Neisseria gonorrhoeae

Neisseria gonorrhoeae data are sourced from Pathogenwatch, which calls AMR and lineage genotypes (MLST, NG-MAST) from genomes assembled from public data. The prevalence estimates shown are calculated using genome collections derived from non-targeted sampling frames (i.e. surveillance and burden studies, as opposed to AMR focused studies or outbreak investigations). These include EuroGASP 2013 & 2018, and several national surveillance studies. Last update: 5 August 2025.

Variable definitions

- **Genotypes**: sequence types from the 7-locus MLST scheme for *Neisseria*, or 2-locus *N. gonorrhoeae* multi-antigen sequence typing (NG-MAST) scheme, both hosted by PubMLST.
- **AMR determinants** are identified by Pathogenwatch using an inhouse dictionary developed and maintained in consultation with an expert advisory group, described here.
- **AMR determinants within genotypes** - This plot shows combinations of determinants that result in clinical resistance to Azithromycin or Ceftriaxone, as defined in Figure 3 of Sánchez-Busó et al (2021).

- **Susceptible to cat I/II drugs** - No determinants found for Azithromycin, Ceftriaxone, Cefixime (category I) or Benzylpenicillin, Ciprofloxacin, Spectinomycin (category II).

Abbreviations

- **MDR**: multidrug resistant (Resistant to one of Azithromycin / Ceftriaxone / Cefixime [category I representatives], plus two or more of Benzylpenicillin / Ciprofloxacin / Spectinomycin [category II representatives])
- **XDR**: extensively drug resistant (Resistant to two of Azithromycin / Ceftriaxone / Cefixime [category I representatives], plus three of Benzylpenicillin / Ciprofloxacin / Spectinomycin [category II representatives])

Note

These definitions are based on those defined in the European CDC Response Plan, modified to use the specific representatives of category I and II antibiotic classes that are available in the dashboard.

Data source

The data are collated from studies with the following PubMed IDs:

Study accession
PRJEB58139
PRJEB4024
PRJEB4024
PRJNA266539
PRJNA298332
PRJEB7904
PRJNA348107
PRJEB23008
PRJEB14168
PRJNA209340, PRJNA209376, PRJNA209345, PRJNA209319 PRJNA209352, PRJNA209333, PRJNA209347, PRJNA209316 PRJ
PRJEB17738
PRJEB10104
PRJDB6504,PRJDB6496
PRJEB34425
PRJEB10016
PRJNA392203
PRJNA473385
PRJEB19989
PRJEB9227
PRJNA394216
PRJNA317462
PRJEB14933
PRJEB32435
PRJNA315363
PRJEB34425
PRJNA317462, PRJNA329501
PRJEB32435
PRJNA520805
PRJEB34068
PRJEB2999

Tab

Study accession
PRJEB32435
PRJEB49317
PRJEB47922
PRJEB50649
PRJEB36607
PRJEB36608
PRJEB40983
PRJEB41007
PRJNA315363
PRJNA317462
PRJNA329501
PRJNA278367
PRJNA481622
PRJNA575338
PRJNA594714
PRJNA664319
PRJNA681740
PRJNA684048
PRJNA738299
PRJNA522696
PRJNA776899
PRJNA785548

Salmonella (non-typhoidal)

Nontyphoidal *Salmonella* data in AMRnet are drawn from Enterobase, which calls AMR genotypes using NCBI's AMRFinderPlus, assigns lineages using MLST, cgMLST and hierarchical clustering, and assigns serotypes using SISTR. The *Salmonella* (non-typhoidal) dashboard includes all genomes classified as *S. enterica* subsp. *enterica* (based on hierarchical clustering level 2850, HC2850=2), and excluding all genomes with predicted serotype Typhi or Paratyphi A/B/C. Last update: 5 August 2025.

For information about *Salmonella* Typhi, please see the Typhi dashboard. Inclusion of Paratyphi A is planned for 2026.

Warning

The non-typhoidal *Salmonella* data used in AMRnet are not yet curated for purpose-of-sampling, and therefore reflect the biases of global sequencing efforts which may be skewed towards sequencing AMR strains and/or outbreaks. Data curation efforts are ongoing however until then, please be careful when interpreting the data in the dashboard.

Variable definitions

- **Lineages** - Lineages are labeled by 7-locus sequence type (ST).
- **AMR determinants:** - Enterobase identifies AMR determinants using NCBI's AMRFinderPlus run with default settings. AMRnet imports these AMR genotype calls, and assigns them to drugs/classes in the dashboard using the Subclass curated in refgenes, see *the table*.

Salmonella (invasive non-typhoidal)

Salmonella (invasive non-typhoidal) data in AMRnet are drawn from Enterobase, which calls AMR genotypes using NCBI's AMRFinderPlus, assigns lineages using MLST, cgMLST and hierarchical clustering, and assigns serotypes using SISTR. The iNTS dashboard currently includes all genomes identified as serotype Typhimurium or Enteritidis (which account for >90% of iNTS), and identifies lineages thereof using MLST. Last update: 5 August 2025.

The invasive non-typhoidal *Salmonella* (iNTS) dashboard is populated with data from specific *Salmonella enterica* lineages that are associated with invasive disease in low-income countries; namely serotype Typhimurium (ST19, ST313 and sublineages thereof as defined by Van Puyvelde et al) and Enteritidis (Central/Eastern and West African clades as defined by Fong et al). Together these account for >90% of iNTS.

Warning

The iNTS data used in AMRnet are not yet curated for purpose-of-sampling, and therefore reflect the biases of global sequencing efforts which may be skewed towards sequencing AMR strains and/or outbreaks. Data curation efforts are ongoing however until then, please be careful when interpreting the data in the dashboard.

Variable definitions

- **Lineages**: Lineages are labeled by iTYM (invasive Typhimurium) or iENT (invasive Enteritidis) followed by the lineage name, defined from cgMLST as follows:
  - iTYM ST19-L1: HC150=305
  - iTYM ST19-L3: HC150=1547
  - iTYM ST19-L4: HC150=48
  - iTYM ST313-L1: HC150=9882
  - iTYM ST313-L2: HC150=728 and HC50=728
  - iENT CEAC: HC150=12675
  - iENT WAC: HC150=2452
- **AMR determinants**: Enterobase identifies AMR determinants using NCBI's AMRFinderPlus run with default settings. AMRnet imports these AMR genotype calls, and assigns them to drugs/classes in the dashboard using the Subclass curated in refgenes, see *the table*.

Salmonella Typhi

Salmonella Typhi data in AMRnet are drawn from Pathogenwatch, which calls AMR determinants and GenoTyphi genotypes from genome assemblies. The *Salmonella* Typhi data in Pathogenwatch are curated by the Global Typhoid Genomics Consortium, as described here.

The prevalence estimates shown are calculated using genome collections derived from non-targeted sampling frames (i.e. surveillance and burden studies, as opposed to AMR focused studies or outbreak investigations) with known year of isolation and country of origin. Last update: 5 August 2025.

Variable definitions

- **Genotypes**: GenoTyphi scheme, see Dyson & Holt, 2021.
- **AMR determinants** and technical details of how they are called are described in the Typhi Pathogenwatch paper and documentation.
- **Travel-associated cases** are attributed to the country of travel, not the country of isolation, see Ingle et al, 2019.

Abbreviations

- **MDR**: multidrug resistant (resistant to ampicillin, chloramphenicol, and trimethoprim-sulfamethoxazole)
- **XDR**: extensively drug resistant (MDR plus resistant to ciprofloxacin and ceftriaxone)
- **Ciprofloxacin NS**: ciprofloxacin non-susceptible (MIC >=0.06 mg/L, due to presence of one or more *qnr* genes or mutations in *gyrA/parC/gyrB*)
- **Ciprofloxacin R**: ciprofloxacin resistant (MIC >=0.5 mg/L, due to presence of multiple mutations and/or genes)

Related tools

The AMRnet project is based on the TyphiNET dashboard, available at https://www.typhi.net/ and described in the paper: Dyson et al, Genome Medicine 2025. The TyphiNET dashboard includes some different visualisations to AMRnet, and is maintained in the form described in the TyphiNET paper, in parallel to AMRnet and populated with the same underlying data as curated by the GTGC. Notably, TyphiNET has the additional feature of plotting combinations of AMR markers (for a single drug/class), per genotype.

2.3 Source data & tools

2.3.1 Source Databases

Enterobase

- EnteroBase in 2025: exploring the genomic epidemiology of bacterial pathogens, Dyer et al 2025
- https://enterobase.warwick.ac.uk/

Pathogenwatch

- *Salmonella* Typhi: A global resource for genomic predictions of antimicrobial resistance and surveillance of Salmonella Typhi at pathogenwatch, Argimón et al 2021
- *Neisseria gonorrhoeae*: A community-driven resource for genomic epidemiology and antimicrobial resistance prediction of Neisseria gonorrhoeae at Pathogenwatch, Sánchez-Busó et al 2021
- *Klebsiella pneumoniae*: Rapid Genomic Characterization and Global Surveillance of Klebsiella Using Pathogenwatch, Argímon 2021
- https://pathogen.watch/

2.3.2 Typing tools

AMRFinderPlus

Used for AMR typing in *E. coli/Shigella* and non-typhoidal *Salmonella* dashboards

- AMRFinderPlus: https://github.com/ncbi/amr
- AMRFinderPlus paper: AMRFinderPlus and the Reference Gene Catalog facilitate examination of the genomic links among antimicrobial resistance, stress response, and virulence, Feldgarden 2021
- Integration of AMRFinderPlus in Enterobase: EnteroBase in 2025: exploring the genomic epidemiology of bacterial pathogens, Dyer et al 2025

Escherichia coli and *Shigella*

Genotype nomenclature

- Achtman 7-locus MLST scheme: Developed by Enterobase
- Core genome MLST scheme: The EnteroBase user's guide, with case studies on Salmonella transmissions, Yersinia pestis phylogeny, and Escherichia core genomic diversity, Zhou et al 2020
- Hierarchical clustering: HierCC: a multi-level clustering scheme for population assignments based on core genome MLST, Zhou et al 2020
- MLST database: https://enterobase.warwick.ac.uk/species/index/ecoli/

Pathotype definition

- The diarrheagenic *E. coli* (DEC) pathotypes are defined using Enterobase pathotyping logic
- *Shigella* species identification : Hierarchical Clustering (HC)

Salmonella (non-typhoidal)

Genotype nomenclature

- 7-locus MLST scheme: Enterobase
- Core genome MLST scheme: The EnteroBase user's guide, with case studies on Salmonella transmissions, Yersinia pestis phylogeny, and Escherichia core genomic diversity, Zhou et al 2020
- Hierarchical clustering: HierCC: a multi-level clustering scheme for population assignments based on core genome MLST, Zhou et al 2020

Serotyping

- Serotypes: The Salmonella In Silico Typing Resource (SISTR): An Open Web-Accessible Tool for Rapidly Typing and Subtyping Draft Salmonella Genome Assemblies, Yoshida et al 2016

Salmonella Typhi

- GenoTyphi scheme: Five Years of GenoTyphi: Updates to the Global Salmonella Typhi Genotyping Framework, Dyson & Holt, 2021
- AMR prediction: A global resource for genomic predictions of antimicrobial resistance and surveillance of Salmonella Typhi at pathogenwatch, Argimón et al 2021

Neisseria gonorrhoeae

- 7-locus MLST scheme: Species status of Neisseria gonorrhoeae: evolutionary and epidemiological inferences from multilocus sequence typing, Bennett et al 2007
- 2-locus *N. gonorrhoeae* multi-antigen sequence typing (NG-MAST) scheme Rapid Sequence-Based Identification of Gonococcal Transmission Clusters in a Large Metropolitan Area, Martin et al 2004
- MLST and NG-MAST databases: PubMLST
- AMR prediction: A community-driven resource for genomic epidemiology and antimicrobial resistance prediction of Neisseria gonorrhoeae at Pathogenwatch, Sánchez-Busó et al 2021

Klebsiella pneumoniae

Genotype nomenclature

- 7-locus MLST scheme: Multilocus Sequence Typing of Klebsiella pneumoniae Nosocomial Isolates, Diancourt 2005
- Core genome MLST scheme: Genomic Definition of Hypervirulent and Multidrug-Resistant Klebsiella pneumoniae Clonal Groups, Bialek-Davenet et al 2014
- Sublineages: A Dual Barcoding Approach to Bacterial Strain Nomenclature: Genomic Taxonomy of Klebsiella pneumoniae Strains, Hennart et al 2022
- MLST databases: BIGSdb Institut Pasteur

AMR typing

- Kleborate v3: https://github.com/klebgenomics/Kleborate
- Kleborate paper: A genomic surveillance framework and genotyping tool for Klebsiella pneumoniae and its related species complex, Lam et al 2021

2.4 Data access

AMRnet data can be accessed in three ways: (1) directly from the dashboard via the **Download** button at the bottom of each organism page, (2) as flat files from the public S3 bucket, and (3) programmatically through the public REST API at api.amrnet.org.

2.4.1 1. Download via the dashboard

Every organism page in the dashboard has a *Download database (TSV format)* button at the bottom. This returns the full sample-level dataset for the currently-selected organism, including all curated metadata and AMR determinant fields. No registration is required.

2.4.2 2. Download via the public S3 bucket

Up-to-date compressed CSV snapshots of the whole database are published weekly to a public AWS S3 bucket. Choose the organism from the table below and append its key to `https://amrnet.s3.amazonaws.com/amrnet-latest/`.

Organism	S3 key
Salmonella Typhi	`amrnetdb-Salmonella_Typhi.csv.gz`
Salmonella (non-typhoidal)	`amrnetdb-Salmonella_enterica_nontyphoidal.csv.gz`
Salmonella (invasive non-typhoidal)	`amrnetdb-Salmonella_enterica_invasive_nontyphoidal.csv.gz`
Klebsiella pneumoniae	`amrnetdb-Klebsiella_pneumoniae.csv.gz`
Neisseria gonorrhoeae	`amrnetdb-Neisseria_gonorrhoeae.csv.gz`
Escherichia coli	`amrnetdb-Escherichia_coli.csv.gz`
Escherichia coli (diarrheagenic)	`amrnetdb-Escherichia_coli_diarrheagenic.csv.gz`
Shigella + EIEC	`amrnetdb-Shigella_EIEC.csv.gz`

Example — browser or `curl`:

```
# Download the S. Typhi snapshot
curl -O https://amrnet.s3.amazonaws.com/amrnet-latest/amrnetdb-Salmonella_Typhi.csv.gz

# List every file in the bucket (XML response)
curl -s https://amrnet.s3.amazonaws.com/
```

Example — s3cmd (unauthenticated; the bucket is public-read):

```
s3cmd --no-check-certificate get s3://amrnet/amrnet-latest/amrnetdb-Salmonella_Typhi.csv.gz
```

2.4.3 3. Programmatic access: AMRnet REST API

The public REST API at **https://api.amrnet.org** exposes the same curated dataset that powers the dashboard. It replaces the earlier MongoDB Atlas Data API described in previous versions of this document.

3.1 Get an API key

Register a key at https://api.amrnet.org/api-register. Registration is free; a key is emailed to you. All endpoints require the key passed either as:

- HTTP header — `X-API-Key: YOUR_KEY` (recommended)
- Query parameter — `?api_key=YOUR_KEY`

3.2 Interactive docs (Swagger / OpenAPI)

Full interactive documentation lives at https://api.amrnet.org/api-docs. Every endpoint has a **Try it out** button that issues a live request — paste your key into the "Authorize" dialog and the built-in server selector already targets the production host.

The OpenAPI 3.0 spec is available at https://api.amrnet.org/api-docs.json if you want to generate a client library with `openapi-generator`.

3.3 Endpoints

3.4 Quick start — `curl`

```
API_KEY="your-registered-api-key"

# List every organism with its current genome count
curl -H "X-API-Key: $API_KEY" https://api.amrnet.org/api/v1/organisms

# Resistance prevalence for S. enterica in Brazil
curl -H "X-API-Key: $API_KEY" \
  "https://api.amrnet.org/api/v1/organisms/senterica/resistance?country=Brazil"

# Per-country summary for E. coli
curl -H "X-API-Key: $API_KEY" https://api.amrnet.org/api/v1/organisms/ecoli/countries

# Full CSV export for S. Typhi
curl -H "X-API-Key: $API_KEY" \
  "https://api.amrnet.org/api/v1/organisms/styphi/download?format=csv" \
  -o styphi.csv
```

3.5 Quick start — Python

```python
import requests
import pandas as pd

BASE = "https://api.amrnet.org/api/v1"
HEADERS = {"X-API-Key": "your-registered-api-key"}

# Organism list
orgs = requests.get(f"{BASE}/organisms", headers=HEADERS).json()

# Resistance prevalence for S. Typhi
r = requests.get(
    f"{BASE}/organisms/styphi/resistance",
    headers=HEADERS,
    params={"country": "India", "year_from": 2018, "year_to": 2023},
).json()

df = pd.DataFrame(r["data"])
```

3.6 Quick start — R

```r
library(httr)
library(jsonlite)

base <- "https://api.amrnet.org/api/v1"
key  <- "your-registered-api-key"

r <- GET(
  paste0(base, "/organisms/kpneumo/resistance"),
  add_headers(`X-API-Key` = key),
  query = list(country = "Pakistan")
)
df <- fromJSON(content(r, "text"))$data
```

3.7 Rate limits

- `/api/v1/*` — 30 requests / second per IP, bursts of 50
- `/api/v1/organisms/*/download` — 2 requests / second per IP, bursts of 5

These limits are enforced by nginx upstream of the Node application. If you need a bulk export, the S3 snapshots in section 2 are the recommended source — no key, no rate limit, and the file is already compressed.

2.4.4 Citation

If you use AMRnet data in published work, please cite:

Cerdeira LT, Dyson ZA, Sharma V, et al. **AMRnet: a data visualization platform to interactively explore pathogen variants and antimicrobial resistance.** Nucleic Acids Res. 2025. doi: 10.1093/nar/gkaf1101.

Please also cite the upstream data sources relevant to the organism(s) you queried — these are listed on each pathogen page and under *Source data & tools*.

2.5 Developer Guide

2.5.1 Features

AMRnet provides a comprehensive suite of features designed to make antimicrobial resistance (AMR) genomic surveillance data accessible and actionable for researchers, public health professionals, and policymakers worldwide.

Interactive Visualizations

Interactive Global Maps

- Visualize resistance patterns across countries and regions
- Color-coded prevalence data with geographic distribution
- Clickable countries for detailed filtering
- Export capabilities for static maps (PNG format)
- Responsive design that works on all devices

Dynamic Trend Analysis

- Track resistance changes over time with interactive graphs
- Multiple chart types: line charts, bar charts, and scatter plots
- Temporal filtering by year, month, or custom date ranges
- Real-time data updates as filters are applied
- Downloadable chart data and static images

Advanced Data Exploration

- Multi-dimensional filtering by organism, drug, genotype, and geography
- Dynamic legends with click-to-filter functionality
- Cross-chart filtering that updates all visualizations simultaneously
- Custom aggregation options (counts vs. percentages)
- Real-time data validation and error handling

Supported Organisms

Comprehensive Pathogen Coverage

AMRnet supports surveillance data for eight major pathogens:

- **Salmonella Typhi** (*S.* Typhi) - Typhoid fever surveillance with H58 lineage tracking
- **Klebsiella pneumoniae** (*K. pneumoniae*) - Healthcare-associated infections and carbapenemase monitoring
- **Neisseria gonorrhoeae** (*N. gonorrhoeae*) - Gonorrhea resistance monitoring with WHO surveillance data
- **Escherichia coli** (*E. coli*) - ESBL and carbapenemase tracking, STEC surveillance
- **Diarrheagenic E. coli** (*dEcoli*) - Specialized diarrheal disease surveillance
- **Shigella** (*Shigella* spp.) - Dysentery and MDR monitoring with serotype tracking
- **Salmonella enterica** (*S. enterica*) - Non-typhoidal Salmonella surveillance
- **Salmonella enterica INTS** - Invasive non-typhoidal Salmonella tracking

Internationalization

Multi-Language Support

- **English** (Default) - Comprehensive interface language
- **French** - Professional French translations with proper terminology
- **Portuguese (Brazil)** - Brazilian Portuguese with local conventions
- **Spanish** - International Spanish translations
- Visual flag-based language switcher
- Persistent language selection across sessions
- Right-to-left language support ready

User Experience

Responsive Design

- Optimized for desktop, tablet, and mobile devices
- Touch-friendly interface for mobile users
- Adaptive layouts that work across screen sizes
- Progressive web app capabilities
- Offline data caching for improved performance

Performance Optimization

- Lazy loading of data and components
- Intelligent caching to reduce server load
- Compressed data transfer (up to 96% size reduction)
- Parallel data loading for faster initial page loads
- Background processing for large datasets

Data Access & Export

Flexible Data Export

- Download filtered datasets in CSV format
- Export static visualizations (PNG, SVG)
- Generate PDF reports with current dashboard state
- Custom data selection with field-level control
- Batch export capabilities for multiple organisms

API Access

- RESTful API with comprehensive documentation
- Programmatic access to all dashboard data
- Rate limiting and authentication for production use
- OpenAPI/Swagger documentation
- SDK libraries for common programming languages

Advanced Features

Scientific Analysis Tools

- Statistical trend analysis with confidence intervals
- Geographic clustering and hotspot identification
- Phylogenetic data integration where available
- Resistance mechanism tracking (genotypic vs. phenotypic)
- Outbreak detection and monitoring capabilities

Data Quality & Security

- Comprehensive data validation and quality checks
- Secure data transmission with HTTPS encryption
- User authentication and authorization
- Audit logging for data access and modifications
- Privacy-compliant data handling procedures

Real-time Updates

- Automatic data synchronization from surveillance networks
- Version control for datasets with change tracking
- Notification system for data updates
- Backward compatibility for historical data access
- Data provenance tracking and metadata management

Integration Capabilities

External System Integration

- NCBI pathogen database connectivity
- WHO surveillance network integration
- ECDC/CDC data pipeline compatibility
- Laboratory information system (LIS) connectors
- Automated data validation and quality scoring

Analytics & Reporting

- Customizable dashboard views for different user types
- Automated report generation with scheduling
- Key performance indicator (KPI) tracking
- Comparative analysis between regions and time periods
- Statistical significance testing for trend analysis

2.5.2 Performance Optimization

AMRnet has been extensively optimized to handle large genomic surveillance datasets efficiently. This guide covers performance considerations, optimization strategies, and deployment best practices.

Architecture Overview

AMRnet's performance optimizations span the entire application stack:

Frontend Optimizations:

- **Lazy Loading**: Components and data loaded on-demand
- **Virtual Scrolling**: Efficient rendering of large datasets
- **Intelligent Caching**: Browser-based storage for frequently accessed data
- **Compression**: Gzipped responses reduce transfer by up to 96%
- **Progressive Loading**: Priority-based data fetching

Backend Optimizations:

- **Database Indexing**: Optimized MongoDB indexes for common queries
- **Aggregation Pipelines**: Server-side data processing
- **Connection Pooling**: Efficient database connection management
- **Field Projection**: Transfer only necessary data fields
- **Parallel Processing**: Concurrent data processing for multiple requests

Load Time Improvements

Recent optimizations have dramatically improved load times across all organisms:

Performance Benchmarks:

Table 2: Load Time Improvements

Organism	Before (seconds)	After (seconds)	Improvement
K. pneumoniae	7-8s (63MB)	1.4s (2.3MB)	81% faster
E. coli	21-23s (186MB)	7s (13MB)	70% faster
E. coli (diarrheagenic)	6-15s (51MB)	2s (4MB)	87% faster

Optimization Strategies:

1. **Optimized Endpoints**: `/api/optimized/*` routes with minimal payloads
2. **Pagination**: Large datasets split into manageable chunks
3. **Parallel Loading**: Multiple chart sections loaded simultaneously
4. **Smart Caching**: Frequently accessed data cached locally

Database Optimization

MongoDB performance optimizations ensure fast query responses:

Indexing Strategy:

```
// Compound indexes for common query patterns
db.ecoli_data.createIndex({ COUNTRY_ONLY: 1, YEAR: 1 })
db.kpneumo_data.createIndex({ GENOTYPE: 1, COUNTRY_ONLY: 1 })
db.styphi_data.createIndex({ GENOTYPE: 1, YEAR: 1, COUNTRY_ONLY: 1 })
```

Aggregation Pipelines:

```
// Efficient data aggregation with field projection
db.collection.aggregate([
  { $match: { COUNTRY_ONLY: "BGD", YEAR: { $gte: 2020 } } },
  { $project: {
      COUNTRY_ONLY: 1,
      YEAR: 1,
      GENOTYPE: 1,
      RESISTANCE_PROFILE: 1,
      _id: 0
    } },
  { $group: {
      _id: { country: "$COUNTRY_ONLY", year: "$YEAR" },
      count: { $sum: 1 },
      genotypes: { $addToSet: "$GENOTYPE" }
    } }
])
```

Connection Optimization:

```
// MongoDB connection settings for production
const mongoOptions = {
  maxPoolSize: 10,
  minPoolSize: 5,
  maxIdleTimeMS: 30000,
  serverSelectionTimeoutMS: 5000,
  socketTimeoutMS: 45000,
  bufferMaxEntries: 0
};
```

Frontend Performance

React application optimizations for smooth user experience:

Component Optimization:

```
// Memoized components prevent unnecessary re-renders
import React, { memo, useMemo, useCallback } from 'react';

const OptimizedChart = memo(({ data, filters }) => {
  const processedData = useMemo(() =>
    processChartData(data, filters), [data, filters]
  );
```

(continues on next page)

(continued from previous page)

```
  const handleFilterChange = useCallback((newFilter) => {
    // Debounced filter updates
    debounce(() => updateFilters(newFilter), 300);
  }, []);

  return <Chart data={processedData} onFilterChange={handleFilterChange} />;
});
```

Data Loading Optimization:

```
// Parallel data loading with Promise.all
const loadOrganismData = async (organism) => {
  const [mapData, trendsData, resistanceData] = await Promise.all([
    fetch(`/api/optimized/map/${organism}`),
    fetch(`/api/optimized/trends/${organism}`),
    fetch(`/api/optimized/resistance/${organism}`)
  ]);

  return {
    map: await mapData.json(),
    trends: await trendsData.json(),
    resistance: await resistanceData.json()
  };
};
```

Virtual Scrolling for Large Lists:

```
import { FixedSizeList as List } from 'react-window';

const LargeDataList = ({ data }) => (
  <List
    height={400}
    itemCount={data.length}
    itemSize={50}
    itemData={data}
  >
    {({ index, style, data }) => (
      <div style={style}>
        {/* Render only visible items */}
        <DataRow item={data[index]} />
      </div>
    )}
  </List>
);
```

Deployment Optimization

Production deployment configurations for optimal performance:

Heroku Configuration:

```
# Environment variables for production
NODE_ENV=production
MONGODB_URI=mongodb+srv://...

# Enable compression
ENABLE_COMPRESSION=true

# Connection pooling
DB_POOL_SIZE=10

# Cache settings
CACHE_TTL=300
```

CDN Integration:

```
// Static asset optimization
const nextConfig = {
  images: {
    domains: ['cdn.amrnet.org'],
    formats: ['image/webp', 'image/avif'],
  },
  compiler: {
    removeConsole: process.env.NODE_ENV === 'production',
  },
  experimental: {
    optimizeCss: true,
  }
};
```

Monitoring and Alerting:

```
// Performance monitoring
const performanceMonitor = {
  trackPageLoad: (pageName, loadTime) => {
    if (loadTime > 3000) {
      console.warn(`Slow page load: ${pageName} took ${loadTime}ms`);
    }
  },

  trackAPICall: (endpoint, responseTime, payloadSize) => {
    if (responseTime > 2000 || payloadSize > 5000000) {
      console.warn(`Performance issue: ${endpoint}`);
    }
  }
};
```

Caching Strategies

Multi-level caching for optimal performance:

Browser Caching:

```
// Service worker for offline capabilities
const CACHE_NAME = 'amrnet-v1';
```

(continues on next page)

(continued from previous page)

```
const urlsToCache = [
  '/',
  '/static/css/main.css',
  '/static/js/main.js',
  '/api/metadata'
];

self.addEventListener('fetch', event => {
  event.respondWith(
    caches.match(event.request)
      .then(response => response || fetch(event.request))
  );
});
```

Redis Caching (Optional):

```
// Server-side caching for frequently accessed data
const redis = require('redis');
const client = redis.createClient(process.env.REDIS_URL);

const getCachedData = async (key) => {
  const cached = await client.get(key);
  return cached ? JSON.parse(cached) : null;
};

const setCachedData = async (key, data, ttl = 300) => {
  await client.setex(key, ttl, JSON.stringify(data));
};
```

Performance Monitoring

Real-time performance tracking and optimization:

Client-Side Metrics:

```
// Web Vitals monitoring
import { getCLS, getFID, getFCP, getLCP, getTTFB } from 'web-vitals';

const sendToAnalytics = (metric) => {
  // Send performance metrics to monitoring service
  fetch('/api/analytics', {
    method: 'POST',
    body: JSON.stringify(metric)
  });
};

getCLS(sendToAnalytics);
getFID(sendToAnalytics);
getFCP(sendToAnalytics);
getLCP(sendToAnalytics);
getTTFB(sendToAnalytics);
```

Server-Side Monitoring:

```
// Express middleware for performance tracking
const performanceMiddleware = (req, res, next) => {
  const start = Date.now();

  res.on('finish', () => {
    const duration = Date.now() - start;
    console.log(`${req.method} ${req.path}: ${duration}ms`);

    // Alert if response time exceeds threshold
    if (duration > 2000) {
      console.warn(`Slow request: ${req.path} took ${duration}ms`);
    }
  });

  next();
};
```

Best Practices

Development Best Practices:

1. **Measure First**: Use browser dev tools to identify bottlenecks
2. **Optimize Queries**: Use database explain plans to optimize queries
3. **Monitor Bundle Size**: Keep JavaScript bundles under 250KB
4. **Image Optimization**: Use modern formats (WebP, AVIF) and responsive images
5. **Code Splitting**: Load only necessary code for each page

Production Best Practices:

1. **Enable Compression**: Use gzip/brotli compression
2. **CDN Usage**: Serve static assets from CDN
3. **Database Indexes**: Ensure proper indexing for all queries
4. **Connection Pooling**: Optimize database connection pools
5. **Performance Monitoring**: Set up alerts for performance degradation

Troubleshooting

Common Performance Issues:

1. **Slow Page Loads**: Check network tab for large payloads
2. **High Memory Usage**: Use Chrome DevTools Memory tab
3. **Database Timeouts**: Review MongoDB slow query logs
4. **Cache Misses**: Verify cache configuration and TTL settings

Performance Testing:

```
# Load testing with Artillery
npm install -g artillery
artillery quick --count 100 --num 10 https://amrnet.org
```

(continues on next page)

(continued from previous page)

```
# Bundle analysis
npm run build
npm run analyze
```

Monitoring Tools:

- Browser DevTools (Performance tab)
- MongoDB Compass (Query performance)
- Heroku metrics (if deployed on Heroku)
- Web Vitals extension
- Lighthouse CI for automated testing

2.5.3 Deployment Guide

This guide covers deployment strategies for AMRnet, including local development, staging environments, and production deployment on various platforms.

Environment Setup

AMRnet supports multiple deployment environments with different configurations:

Development Environment

For local development with hot reloading and debugging:

```
# Clone and setup
git clone https://github.com/amrnet/amrnet.git
cd amrnet

# Install dependencies
npm install
cd client && npm install && cd ..

# Environment configuration
cp .env.example .env
```

Environment Variables (.env):

```
NODE_ENV=development
PORT=8080

# Database
MONGODB_URI=mongodb://localhost:27017/amrnet

# Development settings
REACT_APP_API_URL=http://localhost:8080/api/
ENABLE_DEBUG_LOGS=true
```

Start Development Servers:

```
# Start both backend and frontend
npm run start:dev

# Or individually:
npm run start:backend  # Backend only (port 8080)
npm run client         # Frontend only (port 3000)
```

Staging Environment

For testing production builds locally:

```
# Build production bundle
npm run build

# Start production server
NODE_ENV=production npm start
```

Production Deployment

Heroku Deployment

AMRnet is optimized for Heroku deployment with automatic build processes:

1. Heroku App Setup:

```
# Install Heroku CLI and login
heroku login

# Create new app
heroku create your-app-name

# Provision MongoDB (e.g., mLab sandbox or an external managed cluster).
# AMRnet production itself runs MongoDB on the same EC2 host as the app;
# Heroku is shown here as an alternative hosting path.
heroku addons:create mongolab:sandbox
```

2. Environment Configuration:

```
# Set environment variables
heroku config:set NODE_ENV=production
heroku config:set MONGODB_URI="your-mongodb-connection-string"
heroku config:set REACT_APP_API_URL="https://your-app-name.herokuapp.com/api/"
```

3. Deployment:

```
# Deploy to Heroku
git add .
git commit -m "Deploy to Heroku"
git push heroku main

# Monitor deployment
heroku logs --tail
```

Heroku Configuration Files:

Procfile:

```
web: node server.js
```

package.json (heroku-postbuild script):

```
{
  "scripts": {
    "heroku-postbuild": "cd client && npm install && npm run build"
  }
}
```

MongoDB on EC2 Configuration

AMRnet production runs MongoDB locally on the application EC2 instance (bound to `127.0.0.1` — no public listener). This keeps the data plane off the public internet and avoids the cost and IP-allowlist maintenance of a managed Atlas cluster.

1. Host Setup:

- Install MongoDB 6.0+ on the EC2 instance
- Keep `bindIp: 127.0.0.1` in `/etc/mongod.conf`
- Enable authentication and create a read/write user for AMRnet
- Store the URI in `/opt/amrnet/.credentials` with `chmod 600`

2. Connection Configuration:

```
# Production .env (EC2) - loopback only
MONGODB_URI=mongodb://amrnet_user:password@127.0.0.1:27017/amrnet?authSource=admin
```

3. MongoDB Compass access (developer laptops):

```
# Open an SSH tunnel, then point Compass at localhost:27018
ssh -L 27018:127.0.0.1:27017 ec2-user@<host>
# See deploy/mongo-tunnel.sh for the wrapper script
```

3. Production Optimizations:

```
// config/db.js - Production MongoDB settings
const mongoOptions = {
  useNewUrlParser: true,
  useUnifiedTopology: true,
  maxPoolSize: 10,
  minPoolSize: 5,
  maxIdleTimeMS: 30000,
  serverSelectionTimeoutMS: 5000,
  socketTimeoutMS: 45000,
  bufferMaxEntries: 0,
  bufferCommands: false,
};
```

Docker Deployment

For containerized deployment:

Dockerfile:

```
# Multi-stage build for optimized production image
FROM node:18-alpine AS builder

WORKDIR /app
COPY package*.json ./
RUN npm ci --only=production

# Build client
COPY client/package*.json ./client/
WORKDIR /app/client
RUN npm ci --only=production
COPY client/ .
RUN npm run build

# Production stage
FROM node:18-alpine AS production

WORKDIR /app
COPY --from=builder /app/node_modules ./node_modules
COPY --from=builder /app/client/build ./client/build
COPY . .

EXPOSE 8080
CMD ["node", "server.js"]
```

docker-compose.yml:

```
version: '3.8'
services:
  amrnet:
    build: .
    ports:
      - "8080:8080"
    environment:
      - NODE_ENV=production
      - MONGODB_URI=mongodb://mongo:27017/amrnet
    depends_on:
      - mongo

  mongo:
    image: mongo:6
    ports:
      - "27017:27017"
    volumes:
      - mongo_data:/data/db

volumes:
  mongo_data:
```

Deployment Commands:

```
# Build and start containers
docker-compose up -d

# View logs
docker-compose logs -f amrnet
```

AWS Deployment

For AWS deployment using Elastic Beanstalk:

1. EB CLI Setup:

```
# Install EB CLI
pip install awsebcli

# Initialize EB application
eb init amrnet

# Create environment
eb create amrnet-production
```

2. Configuration Files:

.ebextensions/01_node_command.config:

```
option_settings:
  aws:elasticbeanstalk:container:nodejs:
    NodeCommand: "node server.js"
  aws:elasticbeanstalk:application:environment:
    NODE_ENV: production
```

3. Deploy:

```
# Deploy to AWS
eb deploy

# Monitor health
eb health
```

Performance Optimization

Production deployment optimizations for better performance:

Build Optimizations

Client Build Configuration:

```
// client/.env.production
GENERATE_SOURCEMAP=false
REACT_APP_NODE_ENV=production

// Build optimizations in package.json
```

(continues on next page)

(continued from previous page)

```
{
  "scripts": {
    "build": "react-scripts build && npm run compress",
    "compress": "gzip -k build/static/js/*.js && gzip -k build/static/css/*.css"
  }
}
```

Server Optimizations

Express.js Production Configuration:

```
// server.js production settings
const express = require('express');
const compression = require('compression');
const helmet = require('helmet');

const app = express();

// Security middleware
app.use(helmet());

// Compression middleware
app.use(compression({
  level: 6,
  threshold: 1024,
}));

// Static file caching
app.use(express.static('client/build', {
  maxAge: '1y',
  etag: false
}));
```

Database Optimizations

MongoDB Production Indexes:

```
// Database indexes for production
db.ecoli_data.createIndex({ COUNTRY_ONLY: 1, YEAR: 1 });
db.kpneumo_data.createIndex({ GENOTYPE: 1, COUNTRY_ONLY: 1 });
db.styphi_data.createIndex({ GENOTYPE: 1, YEAR: 1, COUNTRY_ONLY: 1 });
```

Monitoring and Logging

Production monitoring setup for performance and error tracking:

Application Monitoring

Winston Logging Configuration:

```
// config/logger.js
const winston = require('winston');

const logger = winston.createLogger({
  level: process.env.LOG_LEVEL || 'info',
  format: winston.format.combine(
    winston.format.timestamp(),
    winston.format.errors({ stack: true }),
    winston.format.json()
  ),
  defaultMeta: { service: 'amrnet' },
  transports: [
    new winston.transports.File({ filename: 'logs/error.log', level: 'error' }),
    new winston.transports.File({ filename: 'logs/combined.log' }),
  ],
});

if (process.env.NODE_ENV !== 'production') {
  logger.add(new winston.transports.Console({
    format: winston.format.simple()
  }));
}
```

Performance Monitoring Middleware:

```
// middleware/performance.js
const performanceMiddleware = (req, res, next) => {
  const start = Date.now();

  res.on('finish', () => {
    const duration = Date.now() - start;
    logger.info(`${req.method} ${req.path}`, {
      duration,
      statusCode: res.statusCode,
      ip: req.ip
    });

    // Alert on slow requests
    if (duration > 2000) {
      logger.warn(`Slow request detected: ${req.path} took ${duration}ms`);
    }
  });

  next();
};
```

Error Tracking

Sentry Integration:

```
// Error tracking with Sentry
const Sentry = require('@sentry/node');
```

(continues on next page)

(continued from previous page)

```
Sentry.init({
  dsn: process.env.SENTRY_DSN,
  environment: process.env.NODE_ENV,
});

// Error handler middleware
app.use(Sentry.Handlers.errorHandler());
```

Health Checks

Application Health Endpoint:

```
// Health check endpoint
app.get('/health', async (req, res) => {
  try {
    // Check database connection
    await mongoose.connection.db.admin().ping();

    res.status(200).json({
      status: 'healthy',
      timestamp: new Date().toISOString(),
      uptime: process.uptime(),
      memory: process.memoryUsage(),
      database: 'connected'
    });
  } catch (error) {
    res.status(503).json({
      status: 'unhealthy',
      error: error.message
    });
  }
});
```

Backup and Recovery

Data backup strategies for production environments:

Database Backups

EC2 Snapshot + mongodump Strategy:

- Take AWS EBS snapshots of the MongoDB data volume on a schedule
- Run periodic `mongodump` to an S3 bucket for logical backups
- Test restoration into a staging instance regularly

Manual Backup Scripts:

```
#!/bin/bash
# backup-script.sh

DATE=$(date +%Y%m%d_%H%M%S)
BACKUP_DIR="/backups/amrnet_$DATE"
```

(continues on next page)

(continued from previous page)

```
# Create backup directory
mkdir -p $BACKUP_DIR

# Backup each collection
mongodump --uri="$MONGODB_URI" --out=$BACKUP_DIR

# Compress backup
tar -czf "$BACKUP_DIR.tar.gz" -C /backups "amrnet_$DATE"

# Clean up uncompressed backup
rm -rf $BACKUP_DIR

# Upload to cloud storage (optional)
aws s3 cp "$BACKUP_DIR.tar.gz" s3://amrnet-backups/
```

Application Backups

Code and Configuration Backup:

```
# Git-based backup strategy
git tag -a "production-$(date +%Y%m%d)" -m "Production backup $(date)"
git push origin --tags
```

Security Considerations

Security best practices for production deployment:

Environment Security

Secure Environment Variables:

```
# Use secure credential management
heroku config:set MONGODB_URI="$(cat mongodb_uri.txt)"

# Rotate credentials regularly
heroku config:set SESSION_SECRET="$(openssl rand -base64 32)"
```

Network Security:

```
// CORS configuration
const cors = require('cors');

app.use(cors({
  origin: process.env.ALLOWED_ORIGINS?.split(',') || 'https://amrnet.org',
  credentials: true,
  optionsSuccessStatus: 200
}));
```

Database Security

MongoDB Security:

- Enable authentication and authorization
- Use SSL/TLS for connections
- Implement IP whitelisting
- Regular security updates
- Audit logging for database access

Troubleshooting

Common deployment issues and solutions:

Build Failures:

```
# Clear build cache
rm -rf node_modules package-lock.json
npm install

# Frontend build issues
cd client
rm -rf node_modules package-lock.json build
npm install
npm run build
```

Database Connection Issues:

```
# Test MongoDB connection
mongosh "your-mongodb-uri"

# Check network connectivity
ping cluster.mongodb.net
```

Performance Issues:

```
# Monitor resource usage
heroku ps:exec
top

# Check logs for errors
heroku logs --tail
```

Memory Issues:

```
# Increase Heroku dyno size
heroku ps:scale web=1:standard-2x

# Check memory usage patterns
heroku logs --source=heroku.router
```

2.5.4 Security Guide

AMRnet implements comprehensive security measures to protect user data and ensure the integrity of antimicrobial resistance surveillance information. This guide covers security policies, best practices, and implementation details.

Security Policy

AMRnet follows industry-standard security practices for web applications handling sensitive healthcare and research data.

Vulnerability Reporting

Responsible Disclosure Process

If you discover a security vulnerability, please follow these steps:

1. **DO NOT** create a public GitHub issue
2. **Email security report** to: amrnetdashboard@gmail.com
3. **Include detailed information**: - Description of the vulnerability - Steps to reproduce the issue - Potential impact assessment - Suggested fixes (if available)

Response Timeline:

- **24 hours**: Initial acknowledgment
- **72 hours**: Preliminary assessment and severity classification
- **7 days**: Detailed response with fix timeline
- **30 days**: Target resolution for critical vulnerabilities

Supported Versions

Security updates are provided for:

Table 3: Version Support

Version	Supported
1.1.x	Yes
1.0.x	Yes
< 1.0	No

Application Security

AMRnet implements multiple layers of security controls:

Environment Security

Environment Variable Protection:

```
# Never commit sensitive environment variables
# Use .env.example as template, create .env locally

# Production secrets management
NODE_ENV=production
MONGODB_URI=mongodb+srv://[REDACTED]
```

(continues on next page)

(continued from previous page)

```
SESSION_SECRET=[GENERATED_SECRET]

# Security headers
ENABLE_SECURITY_HEADERS=true
CORS_ORIGIN=https://amrnet.org
```

Secure Configuration Example:

```
// config/security.js
const securityConfig = {
  // HTTPS enforcement
  httpsOnly: process.env.NODE_ENV === 'production',

  // Security headers
  contentSecurityPolicy: {
    directives: {
      defaultSrc: ["'self'"],
      styleSrc: ["'self'", "'unsafe-inline'", "https://fonts.googleapis.com"],
      fontSrc: ["'self'", "https://fonts.gstatic.com"],
      imgSrc: ["'self'", "data:", "https:"],
      scriptSrc: ["'self'"]
    }
  },

  // Rate limiting
  rateLimit: {
    windowMs: 15 * 60 * 1000, // 15 minutes
    max: 1000 // limit each IP to 1000 requests per windowMs
  }
};
```

Database Security

MongoDB Security Configuration:

```
// Secure MongoDB connection
const mongoOptions = {
  useNewUrlParser: true,
  useUnifiedTopology: true,

  // Authentication
  authSource: 'admin',

  // SSL/TLS
  ssl: true,
  sslValidate: true,

  // Connection limits
  maxPoolSize: 10,
  minPoolSize: 5,

  // Timeouts
```

(continues on next page)

(continued from previous page)

```
  serverSelectionTimeoutMS: 5000,
  socketTimeoutMS: 45000,

  // Security options
  bufferMaxEntries: 0,
  bufferCommands: false
};
```

Data Sanitization:

```
const sanitize = require('mongo-sanitize');
const validator = require('validator');

// Input sanitization middleware
const sanitizeInput = (req, res, next) => {
  // Sanitize against NoSQL injection
  req.body = sanitize(req.body);
  req.query = sanitize(req.query);
  req.params = sanitize(req.params);

  // Additional validation
  Object.keys(req.query).forEach(key => {
    if (typeof req.query[key] === 'string') {
      req.query[key] = validator.escape(req.query[key]);
    }
  });

  next();
};
```

API Security

Authentication and Authorization:

```
const jwt = require('jsonwebtoken');
const rateLimit = require('express-rate-limit');

// API rate limiting
const apiLimiter = rateLimit({
  windowMs: 15 * 60 * 1000, // 15 minutes
  max: 1000, // limit each IP to 1000 requests per windowMs
  message: 'Too many requests from this IP',
  standardHeaders: true,
  legacyHeaders: false,
});

// JWT authentication for protected endpoints
const authenticateToken = (req, res, next) => {
  const authHeader = req.headers['authorization'];
  const token = authHeader && authHeader.split(' ')[1];

  if (!token) {
```

(continues on next page)

(continued from previous page)

```
    return res.sendStatus(401);
  }

  jwt.verify(token, process.env.ACCESS_TOKEN_SECRET, (err, user) => {
    if (err) return res.sendStatus(403);
    req.user = user;
    next();
  });
};
```

CORS Configuration:

```
const cors = require('cors');

const corsOptions = {
  origin: function (origin, callback) {
    const allowedOrigins = [
      'https://amrnet.org',
      'https://www.amrnet.org',
      process.env.NODE_ENV === 'development' && 'http://localhost:3000'
    ].filter(Boolean);

    if (!origin || allowedOrigins.includes(origin)) {
      callback(null, true);
    } else {
      callback(new Error('Not allowed by CORS'));
    }
  },
  credentials: true,
  optionsSuccessStatus: 200
};
```

Frontend Security

Content Security Policy:

```
<!-- Security headers in HTML -->
<meta http-equiv="Content-Security-Policy"
      content="default-src 'self';
               script-src 'self' 'unsafe-inline';
               style-src 'self' 'unsafe-inline' https://fonts.googleapis.com;
               font-src 'self' https://fonts.gstatic.com;
               img-src 'self' data: https:;">
```

XSS Prevention:

```
import DOMPurify from 'dompurify';

// Sanitize user input before rendering
const SafeHTML = ({ content }) => {
  const cleanHTML = DOMPurify.sanitize(content);
  return <div dangerouslySetInnerHTML={{ __html: cleanHTML }} />;
```

(continues on next page)

(continued from previous page)

```
};

// Input validation
const validateInput = (input) => {
  if (typeof input !== 'string') return false;
  if (input.length > 1000) return false;

  // Check for malicious patterns
  const maliciousPatterns = [
    /<script\b[^<]*(?:(?!<\/script>)<[^<]*)*<\/script>/gi,
    /javascript:/gi,
    /on\w+\s*=/gi
  ];

  return !maliciousPatterns.some(pattern => pattern.test(input));
};
```

Data Privacy

AMRnet implements privacy-by-design principles for handling surveillance data:

Data Classification

Public Data: - Aggregated surveillance statistics - Country-level prevalence data - Publicly available research datasets

Restricted Data: - Individual sample identifiers (when present) - Detailed geographic coordinates - Unpublished research data

Prohibited Data: - Personal health information (PHI) - Patient identifiers - Clinical details beyond resistance patterns

Privacy Controls

Data Minimization:

```
// Example: Remove sensitive fields before transmission
const sanitizeDataForPublic = (data) => {
  return data.map(record => ({
    // Include only necessary fields
    country: record.COUNTRY_ONLY,
    year: record.YEAR,
    genotype: record.GENOTYPE,
    resistance: record.RESISTANCE_PROFILE,
    // Exclude: individual IDs, precise coordinates, etc.
  }));
};
```

Anonymization:

```
// Geographic aggregation for privacy
const aggregateByRegion = (data) => {
  const aggregated = {};

  data.forEach(record => {
```

(continues on next page)

(continued from previous page)

```
    const region = getRegionFromCountry(record.country);
    if (!aggregated[region]) {
      aggregated[region] = {
        count: 0,
        resistanceProfiles: {}
      };
    }

    aggregated[region].count++;
    // Aggregate resistance data without individual records
  });

  return aggregated;
};
```

Secure Development

Security practices for development and deployment:

Code Security

Dependency Management:

```
# Regular security audits
npm audit

# Update vulnerable dependencies
npm audit fix

# Use lock files to prevent supply chain attacks
npm ci  # Use exact versions from package-lock.json
```

Security Linting:

```
# ESLint security plugin
npm install --save-dev eslint-plugin-security

# .eslintrc.js
module.exports = {
  plugins: ['security'],
  extends: ['plugin:security/recommended'],
  rules: {
    'security/detect-object-injection': 'error',
    'security/detect-non-literal-regexp': 'error',
    'security/detect-unsafe-regex': 'error'
  }
};
```

Git Security

Secure Repository Practices:

```
# Git hooks for security
# pre-commit hook
#!/bin/sh

# Check for secrets in commits
git diff --cached --name-only | xargs grep -l "password\|secret\|key\|token" && {
  echo "Potential secret detected! Commit aborted."
  exit 1
}

# Run security linting
npm run lint:security
```

Secrets Management:

```
# .gitignore - Never commit sensitive files
.env
.env.local
.env.development.local
.env.test.local
.env.production.local

# Security credentials
*.pem
*.key
*.crt

# Database dumps
*.sql
*.dump
```

Deployment Security

Security configurations for production deployment:

Server Security

Security Headers:

```
const helmet = require('helmet');

app.use(helmet({
  // Content Security Policy
  contentSecurityPolicy: {
    directives: {
      defaultSrc: ["'self'"],
      styleSrc: ["'self'", "'unsafe-inline'", "https://fonts.googleapis.com"],
      fontSrc: ["'self'", "https://fonts.gstatic.com"],
      imgSrc: ["'self'", "data:", "https:"],
      scriptSrc: ["'self'"],
```

(continues on next page)

(continued from previous page)

```
      connectSrc: ["'self'", "https://api.amrnet.org"]
    }
  },

  // Other security headers
  hsts: {
    maxAge: 31536000,
    includeSubDomains: true,
    preload: true
  },

  frameguard: { action: 'deny' },
  noSniff: true,
  xssFilter: true,
  referrerPolicy: { policy: 'same-origin' }
}));
```

Infrastructure Security

HTTPS Configuration:

```
# Nginx SSL configuration
server {
    listen 443 ssl http2;
    server_name amrnet.org www.amrnet.org;

    ssl_certificate /path/to/certificate.crt;
    ssl_certificate_key /path/to/private.key;

    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers ECDHE-RSA-AES256-GCM-SHA512:DHE-RSA-AES256-GCM-SHA512;
    ssl_prefer_server_ciphers off;

    # Security headers
    add_header Strict-Transport-Security "max-age=63072000" always;
    add_header X-Frame-Options DENY always;
    add_header X-Content-Type-Options nosniff always;
}
```

Monitoring and Incident Response

Continuous security monitoring and incident response procedures:

Security Monitoring

Logging and Alerting:

```
const winston = require('winston');

// Security event logging
const securityLogger = winston.createLogger({
  level: 'info',
```

(continues on next page)

(continued from previous page)

```
  format: winston.format.combine(
    winston.format.timestamp(),
    winston.format.json()
  ),
  transports: [
    new winston.transports.File({ filename: 'logs/security.log' })
  ]
});

// Failed authentication attempts
const logFailedAuth = (req, ip, reason) => {
  securityLogger.warn('Failed authentication attempt', {
    ip: ip,
    userAgent: req.get('User-Agent'),
    reason: reason,
    timestamp: new Date().toISOString()
  });
};
```

Intrusion Detection:

```
// Suspicious activity detection
const suspiciousActivityDetector = {
  failedAttempts: new Map(),

  checkFailedLogins: (ip) => {
    const attempts = this.failedAttempts.get(ip) || 0;
    this.failedAttempts.set(ip, attempts + 1);

    if (attempts > 5) {
      // Trigger security alert
      this.triggerSecurityAlert(`Multiple failed login attempts from ${ip}`);
    }
  },

  triggerSecurityAlert: (message) => {
    securityLogger.error('Security Alert', { message });
    // Send notification to security team
  }
};
```

Incident Response

Response Procedures:

1. **Detection**: Automated monitoring alerts security team
2. **Assessment**: Determine severity and scope of incident
3. **Containment**: Isolate affected systems if necessary
4. **Investigation**: Analyze logs and determine root cause
5. **Recovery**: Restore normal operations

6. **Lessons Learned**: Update security measures based on findings

Security Testing

Regular security testing ensures ongoing protection:

Automated Testing

Security Test Suite:

```
// __tests__/security.test.js
describe('Security Tests', () => {
  it('should prevent SQL injection in API endpoints', async () => {
    const maliciousInput = "'; DROP TABLE users; --";
    const response = await request(app)
      .get(`/api/organisms?country=${maliciousInput}`)
      .expect(400);

    expect(response.body.error).toContain('Invalid input');
  });

  it('should enforce rate limiting', async () => {
    const requests = Array(1001).fill().map(() =>
      request(app).get('/api/organisms')
    );

    const responses = await Promise.all(requests);
    const rateLimited = responses.filter(r => r.status === 429);
    expect(rateLimited.length).toBeGreaterThan(0);
  });
});
```

Penetration Testing

Regular Security Assessments:

- Quarterly vulnerability scans
- Annual penetration testing by third-party security firms
- Continuous security monitoring
- Bug bounty program for responsible disclosure

Best Practices Checklist

Development Security Checklist:

Environment Security - [ ] Never commit secrets to version control - [ ] Use secure environment variable management - [ ] Implement proper secret rotation

Application Security - [ ] Input validation and sanitization - [ ] Output encoding to prevent XSS - [ ] SQL injection prevention - [ ] Authentication and authorization

Infrastructure Security - [ ] HTTPS enforcement - [ ] Security headers implementation - [ ] Regular dependency updates - [ ] Database access controls

Monitoring and Response - [ ] Security event logging - [ ] Intrusion detection systems - [ ] Incident response procedures - [ ] Regular security assessments

Contact Information

Security Team Contact:

- **Email**: amrnetdashboard@gmail.com
- **PGP Key**: Available upon request
- **Response Time**: 24 hours for initial acknowledgment

For General Security Questions:

- **GitHub Discussions**: https://github.com/amrnet/amrnet/discussions
- **Documentation**: https://amrnet.readthedocs.io

2.5.5 Professional Translation Services

This guide covers integration with professional translation platforms and medical translation services for AMRnet documentation and interface translations.

Overview

AMRnet supports integration with professional translation services to ensure high-quality, medically accurate translations. This is especially important for antimicrobial resistance terminology and clinical interpretations.

> **Note**
>
> Medical translation requires specialized expertise in microbiology, infectious diseases, and clinical laboratory terminology.

Supported Translation Platforms

Weblate Integration

Weblate is an open-source, web-based translation platform ideal for collaborative translation projects.

Setup Configuration

1. **Create Weblate Project**

```
# weblate.yml
project:
  name: "AMRnet"
  slug: "amrnet"
  web: "https://amrnet.org"
  instructions: |
    AMRnet is a surveillance platform for antimicrobial resistance.
    Please ensure medical accuracy when translating clinical terms.

components:
  - name: "Documentation"
    filemask: "docs/locale/*/LC_MESSAGES/*.po"
    template: "docs/locale/en/LC_MESSAGES/template.pot"
    file_format: "po"
    source_language: "en"
```

(continues on next page)

(continued from previous page)

```
  - name: "React Interface"
    filemask: "client/locales/*.json"
    template: "client/locales/en.json"
    file_format: "json"
    source_language: "en"
```

2. **Webhook Integration**

```python
# scripts/weblate_webhook.py
import requests
import subprocess
import os

def handle_weblate_webhook(payload):
    """Handle incoming Weblate webhook for translation updates."""
    if payload.get('event') == 'translation_update':
        component = payload['component']['slug']
        language = payload['translation']['language']['code']

        # Pull latest translations
        subprocess.run([
            'git', 'pull', 'origin',
            f'weblate-{component}-{language}'
        ])

        # Rebuild documentation if needed
        if component == 'documentation':
            subprocess.run(['make', 'html'], cwd='docs/')

        # Update React translations
        if component == 'react-interface':
            subprocess.run(['npm', 'run', 'build'], cwd='client/')
```

3. **Medical Terminology Glossary**

```
# weblate-glossary.yml
glossary:
  - source: "antimicrobial resistance"
    translations:
      es: "resistencia antimicrobiana"
      fr: "résistance antimicrobienne"
      pt: "resist^encia antimicrobiana"
    note: "Primary term for AMR"

  - source: "minimum inhibitory concentration"
    translations:
      es: "concentración inhibitoria mínima"
      fr: "concentration minimale inhibitrice"
      pt: "concentração inibitória mínima"
    note: "MIC - standardized laboratory measurement"
```

Crowdin Integration

Crowdin provides professional translation management with advanced workflows and quality assurance.

Project Setup

```yaml
# crowdin.yml
project_id: "amrnet"
api_token_env: "CROWDIN_API_TOKEN"
base_path: "."
base_url: "https://api.crowdin.com"

preserve_hierarchy: true

files:
  - source: "/docs/locale/en/LC_MESSAGES/*.pot"
    translation: "/docs/locale/%two_letters_code%/LC_MESSAGES/%original_file_name%.po"
    type: "po"

  - source: "/client/locales/en.json"
    translation: "/client/locales/%two_letters_code%.json"
    type: "json"
```

Quality Assurance Configuration

```yaml
# crowdin-qa.yml
quality_assurance:
  checks:
    - "empty_translations"
    - "inconsistent_translations"
    - "missing_translations"
    - "medical_terminology"

custom_checks:
  medical_terminology:
    pattern: "resistance|susceptible|intermediate|MIC|breakpoint"
    message: "Medical term requires specialist review"

workflows:
  - name: "Medical Review"
    steps:
      - type: "translation"
        assignees: ["medical_translators"]
      - type: "proofreading"
        assignees: ["clinical_reviewers"]
      - type: "approval"
        assignees: ["project_managers"]
```

Automated Workflow Integration

```
# scripts/crowdin_automation.py
import crowdin_api
import subprocess
import json

class CrowdinAutomation:
    def __init__(self, api_token, project_id):
        self.client = crowdin_api.Client(api_token)
        self.project_id = project_id

    def upload_source_files(self):
        """Upload source files for translation."""
        # Upload POT files
        pot_files = subprocess.check_output([
            'find', 'docs/locale/en/LC_MESSAGES', '-name', '*.pot'
        ]).decode().strip().split('\n')

        for pot_file in pot_files:
            if pot_file:
                self.client.source_files.upload_file(
                    self.project_id,
                    pot_file,
                    type='po'
                )

    def download_translations(self):
        """Download completed translations."""
        build = self.client.translations.build_project_translation(
            self.project_id
        )

        # Wait for build completion
        while build['data']['status'] != 'finished':
            time.sleep(30)
            build = self.client.translations.check_project_build_status(
                self.project_id, build['data']['id']
            )

        # Download and extract
        self.client.translations.download_project_translations(
            self.project_id, build['data']['id'], 'docs/locale/'
        )
```

Lokalise Integration

Lokalise offers advanced automation and enterprise-grade translation management.

Configuration Setup

```
# lokalise.yml
project_id: "your_project_id"
api_token: "${LOKALISE_API_TOKEN}"

file_mapping:
  documentation:
    file_format: "po"
    original_filenames: true
    directory_prefix: "docs/locale/%LANG_ISO%/LC_MESSAGES/"

  interface:
    file_format: "json"
    json_unescaped_slashes: true
    directory_prefix: "client/locales/"
    filename: "%LANG_ISO%.json"
```

Automated CI/CD Integration

```
# .github/workflows/lokalise-sync.yml
name: Lokalise Translation Sync

on:
  schedule:
    - cron: '0 2 * * *'  # Daily at 2 AM
  workflow_dispatch:

jobs:
  sync-translations:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3

      - name: Install Lokalise CLI
        run: |
          curl -sfL https://raw.githubusercontent.com/lokalise/lokalise-cli-2-go/master/
↪install.sh | sh
          sudo mv ./bin/lokalise2 /usr/local/bin/

      - name: Upload source files
        run: |
          lokalise2 file upload \
            --project-id ${{ secrets.LOKALISE_PROJECT_ID }} \
            --token ${{ secrets.LOKALISE_API_TOKEN }} \
            --file "docs/locale/en/LC_MESSAGES/*.pot" \
            --lang-iso "en"

      - name: Download translations
        run: |
          lokalise2 file download \
            --project-id ${{ secrets.LOKALISE_PROJECT_ID }} \
```

(continues on next page)

(continued from previous page)

```
            --token ${{ secrets.LOKALISE_API_TOKEN }} \
            --format po \
            --unzip-to docs/locale/

      - name: Create Pull Request
        uses: peter-evans/create-pull-request@v4
        with:
          title: "Update translations from Lokalise"
          commit-message: "chore: update translations"
          branch: "translations/update"
```

Medical Translation Services

Specialized Medical Translation

For clinical and regulatory content, consider specialized medical translation services:

```
# medical-translation-workflow.yml
medical_content:
  priority_files:
    - "tutorial.rst"        # Clinical guidance
    - "interpretation.rst"  # Result interpretation
    - "organisms/*.rst"     # Pathogen information

requirements:
  qualifications:
    - "Medical translation certification"
    - "Microbiology/infectious disease background"
    - "ISO 17100 compliance"

  review_process:
    - "Translation by certified medical translator"
    - "Review by clinical microbiologist"
    - "Final approval by AMR specialist"
```

Terminology Management

```
# scripts/medical_terminology.py
import json
import requests

class MedicalTerminologyManager:
    def __init__(self):
        self.terminology_db = self.load_who_terminology()
        self.custom_terms = self.load_custom_glossary()

    def load_who_terminology(self):
        """Load WHO AMR terminology database."""
        # Integration with WHO terminology services
        response = requests.get(
            "https://www.who.int/antimicrobial-resistance/terminology.json"
```

(continues on next page)

(continued from previous page)

```
        )
        return response.json()

    def validate_translation(self, source_term, target_term, language):
        """Validate medical term translation accuracy."""
        if source_term in self.terminology_db:
            official_translation = self.terminology_db[source_term].get(language)
            if official_translation and official_translation != target_term:
                return {
                    "valid": False,
                    "suggestion": official_translation,
                    "source": "WHO terminology"
                }
        return {"valid": True}

    def get_term_context(self, term):
        """Provide context for medical terms."""
        contexts = {
            "MIC": "Minimum Inhibitory Concentration - lowest concentration of␣
↪antimicrobial that inhibits visible growth",
            "EUCAST": "European Committee on Antimicrobial Susceptibility Testing",
            "CLSI": "Clinical and Laboratory Standards Institute"
        }
        return contexts.get(term, "Standard medical term")
```

Quality Assurance Workflows

Automated Quality Checks

```
# scripts/translation_qa.py
import re
import polib
import json

class TranslationQA:
    def __init__(self):
        self.medical_terms = self.load_medical_glossary()
        self.format_patterns = {
            'percentage': r'\d+\.?\d*%',
            'concentration': r'\d+\.?\d*\s*(mg/L|g/mL)',
            'year_range': r'\d{4}-\d{4}'
        }

    def check_po_file(self, po_file_path):
        """Comprehensive QA check for PO files."""
        po_file = polib.pofile(po_file_path)
        issues = []

        for entry in po_file:
            if entry.msgstr:
                issues.extend(self.check_entry(entry))
```

(continues on next page)

(continued from previous page)

```
        return issues

    def check_entry(self, entry):
        """Check individual translation entry."""
        issues = []

        # Check medical terminology consistency
        issues.extend(self.check_medical_terms(entry))

        # Check format preservation
        issues.extend(self.check_format_preservation(entry))

        # Check placeholder preservation
        issues.extend(self.check_placeholders(entry))

        return issues

    def check_medical_terms(self, entry):
        """Verify medical terminology accuracy."""
        issues = []
        source_terms = self.extract_medical_terms(entry.msgid)
        target_terms = self.extract_medical_terms(entry.msgstr)

        for term in source_terms:
            if term in self.medical_terms:
                expected = self.medical_terms[term].get('target_language')
                if expected and expected not in entry.msgstr:
                    issues.append({
                        'type': 'medical_terminology',
                        'term': term,
                        'expected': expected,
                        'location': entry.linenum
                    })

        return issues
```

Integration Scripts

Unified Translation Pipeline

```
# scripts/translation_pipeline.py
import asyncio
import aiohttp
import subprocess
from pathlib import Path

class TranslationPipeline:
    def __init__(self, config):
        self.config = config
        self.platforms = {
            'weblate': WeblateClient(config['weblate']),
            'crowdin': CrowdinClient(config['crowdin']),
```

(continues on next page)

(continued from previous page)

```
            'lokalise': LokaliseClient(config['lokalise'])
        }

    async def sync_all_platforms(self):
        """Synchronize translations across all platforms."""
        tasks = []

        for platform_name, client in self.platforms.items():
            if self.config.get(platform_name, {}).get('enabled'):
                tasks.append(self.sync_platform(platform_name, client))

        results = await asyncio.gather(*tasks, return_exceptions=True)
        return dict(zip(self.platforms.keys(), results))

    async def sync_platform(self, platform_name, client):
        """Sync individual platform."""
        try:
            # Upload source files
            await client.upload_sources()

            # Download completed translations
            translations = await client.download_translations()

            # Run quality checks
            qa_results = self.run_quality_checks(translations)

            return {
                'status': 'success',
                'translations': len(translations),
                'qa_issues': len(qa_results)
            }

        except Exception as e:
            return {
                'status': 'error',
                'error': str(e)
            }
```

Monitoring and Reporting

Translation Progress Dashboard

```
# scripts/translation_dashboard.py
import json
import matplotlib.pyplot as plt
from datetime import datetime

class TranslationDashboard:
    def __init__(self):
        self.platforms = ['weblate', 'crowdin', 'lokalise']
        self.languages = ['es', 'fr', 'pt']
```

(continues on next page)

(continued from previous page)

```
    def generate_progress_report(self):
        """Generate comprehensive translation progress report."""
        report = {
            'generated_at': datetime.now().isoformat(),
            'overall_progress': {},
            'platform_details': {},
            'quality_metrics': {}
        }

        for platform in self.platforms:
            platform_data = self.get_platform_data(platform)
            report['platform_details'][platform] = platform_data

        # Calculate overall progress
        for lang in self.languages:
            progress = self.calculate_language_progress(lang)
            report['overall_progress'][lang] = progress

        # Generate visualizations
        self.create_progress_charts(report)

        return report

    def create_progress_charts(self, report):
        """Create visual progress charts."""
        # Progress by language
        languages = list(report['overall_progress'].keys())
        progress_values = [
            report['overall_progress'][lang]['percentage']
            for lang in languages
        ]

        plt.figure(figsize=(10, 6))
        plt.bar(languages, progress_values)
        plt.title('Translation Progress by Language')
        plt.ylabel('Completion Percentage')
        plt.savefig('docs/_static/translation-progress.png')
        plt.close()
```

Deployment Integration

Read the Docs Configuration

```
# .readthedocs.yml - Enhanced for professional translations
version: 2

build:
  os: ubuntu-22.04
  tools:
    python: "3.11"

sphinx:
```

(continues on next page)

(continued from previous page)

```
  configuration: docs/conf.py
  fail_on_warning: false

formats:
  - pdf
  - epub

python:
  install:
    - requirements: docs/requirements.txt
    - requirements: requirements.txt
    - method: pip
      path: .

# Translation-specific configuration
search:
  ranking:
    api/v2: 2
    tutorial: 10
    guide: 5

# Professional translation webhook
webhooks:
  - url: https://your-domain.com/webhooks/translation-update
    events: [build_success]
```

Best Practices

Translation Workflow Guidelines

1. **Source Content Preparation**
   - Use clear, concise language
   - Avoid idioms and colloquialisms
   - Include context for technical terms
   - Provide glossaries for medical terminology
2. **Translation Quality Standards**
   - Medical accuracy verification
   - Cultural appropriateness review
   - Technical terminology consistency
   - User interface coherence
3. **Review Process**
   - Initial translation by certified translators
   - Medical review by subject matter experts
   - Linguistic review for style and clarity
   - Final approval by project stakeholders

4. **Maintenance and Updates**
    - Regular terminology database updates
    - Continuous quality monitoring
    - User feedback integration
    - Version control for translations

> **Note**
>
> **Security Consideration**: Ensure all translation platforms comply with healthcare data privacy regulations (HIPAA, GDPR) when handling medical content.

Conclusion

Professional translation services integration ensures that AMRnet's critical antimicrobial resistance information is accurately communicated across language barriers. The combination of automated workflows and specialized medical translation expertise provides the quality assurance necessary for clinical decision-making tools.

For implementation support, consult with medical translation specialists familiar with antimicrobial resistance terminology and clinical microbiology practices.

2.5.6 Troubleshooting Guide

This comprehensive troubleshooting guide covers common issues encountered when developing, deploying, and maintaining AMRnet. The solutions are based on real-world problems and their tested fixes.

Performance Issues

Slow Data Loading (2+ Minute Load Times)

Symptoms: - E. coli data taking 3+ minutes to load - K. pneumoniae taking 2+ minutes - Browser freezing during data fetch - Large payload sizes (75MB+)

Diagnosis Script:

```
# Test performance issues
node scripts/organism-performance-debugger.js
```

Root Causes: - Unoptimized MongoDB queries returning all fields - Large datasets (227K+ documents for E. coli) - Single monolithic API calls - Lack of pagination or progressive loading

Solutions:

1. **Use Optimized Endpoints:**

```
// Instead of:
const data = await axios.get('/api/getDataForEcoli'); // 186MB, 21s

// Use:
const data = await axios.get('/api/optimized/map/ecoli'); // 13MB, 7s
```

2. **Implement Pagination for Large Datasets:**

```
// For E. coli (large dataset)
const firstPage = await axios.get('/api/optimized/paginated/ecoli?page=1&limit=5000');

// Progressive loading
const loadNextPage = async (page) => {
  return axios.get(`/api/optimized/paginated/ecoli?page=${page}&limit=5000`);
};
```

3. **Use Parallel Loading:**

```
// Load multiple chart sections simultaneously
const [mapData, trendsData, resistanceData] = await Promise.all([
  axios.get('/api/optimized/map/ecoli'),
  axios.get('/api/optimized/trends/ecoli'),
  axios.get('/api/optimized/resistance/ecoli')
]);
```

Performance Validation:

```
# Test the fixes
node scripts/real-performance-test.js
```

Browser Freezing on Large Datasets

Symptoms: - Browser becomes unresponsive - UI freezes during data processing - "Page Unresponsive" warnings

Diagnosis:

```
# Test for freezing issues
node scripts/ecoli-freeze-debugger.js
```

Solutions:

1. **Progressive Genotype Loading:**

```
// Prevent freezing with requestIdleCallback
const processGenotypesProgressively = (data) => {
  const batches = chunkArray(data, 1000);
  let processed = [];

  const processBatch = (batchIndex) => {
    if (batchIndex >= batches.length) {
      updateUI(processed);
      return;
    }

    requestIdleCallback(() => {
      processed = [...processed, ...batches[batchIndex]];
      processBatch(batchIndex + 1);
    });
  };

  processBatch(0);
};
```

2. **Web Workers for Heavy Processing:**

```
// offload heavy data processing
const worker = new Worker('/workers/data-processor.js');

worker.postMessage({ data: largeDataset });
worker.onmessage = (event) => {
  const processedData = event.data;
  updateUI(processedData);
};
```

Validation:

```
# Verify fix is working
node scripts/post-fix-validation.js
```

Development Issues

ESLint Errors and Warnings

Symptoms: - Build failures due to linting errors - Unused variable warnings - Import/export errors

Quick Fix:

```
# Automated ESLint fixes
./scripts/comprehensive-eslint-fix.sh
```

Manual Fixes:

```
# Fix specific issues
cd client
npx eslint src --fix --max-warnings 200

# For development (bypass errors)
ESLINT_NO_DEV_ERRORS=true npm start
```

Common Issues and Fixes:

```
// Fix unused theme parameters
// Before:
const useStyles = makeStyles((theme) => ({
  root: { padding: 16 }
}));

// After:
const useStyles = makeStyles((_theme) => ({
  root: { padding: 16 }
}));

// Fix unused parameters in catch blocks
// Before:
.catch((error) => console.log('Error occurred'))

// After:
.catch((_error) => console.log('Error occurred'))
```

MongoDB Connection Issues

Symptoms: - "Failed to load resource: 500 Internal Server Error" - Connection timeouts - Authentication failures

Diagnosis:

```
# Test MongoDB connection
node scripts/test-fixie-connection.js
```

Common Solutions:

1. **Check Environment Variables:**

```
# Verify configuration
echo $MONGODB_URI
echo $NODE_ENV
```

2. **Fix Connection String Format:**

```
# Correct format
MONGODB_URI=mongodb+srv://username:password@cluster.mongodb.net/dbname?retryWrites=true&
↪w=majority
```

3. **Test Connection with Retry Logic:**

```
const connectWithRetry = async () => {
  const maxRetries = 3;
  let retries = 0;

  while (retries < maxRetries) {
    try {
      await client.connect();
      return;
    } catch (error) {
      retries++;
      console.log(`Connection attempt ${retries} failed: ${error.message}`);
      if (retries === maxRetries) throw error;
      await new Promise(resolve => setTimeout(resolve, 2000));
    }
  }
};
```

Server Startup Issues

Symptoms: - Server not starting on correct port - "EADDRINUSE" port errors - Import/export syntax errors

Solutions:

1. **Port Configuration:**

```
# Start on specific port
PORT=8080 node server.js
```

2. **Kill Existing Processes:**

```
# Find and kill processes using port
lsof -ti:8080 | xargs kill -9
```

3. **Use Minimal Test Server:**

```
# Start minimal server for testing
node scripts/test-server.js
```

Validation:

```
# Test server endpoints
./scripts/test-api.sh
```

Deployment Issues

Heroku Deployment Failures

Symptoms: - Build failures during deployment - Memory limit exceeded - Slow response times

Pre-deployment Check:

```
# Validate deployment readiness
node scripts/deployment-readiness.js
```

Common Fixes:

1. **Environment Variables:**

```
# Set required variables
heroku config:set NODE_ENV=production
heroku config:set MONGODB_URI="your-connection-string"
```

2. **Optimize for Heroku:**

```
# Run Heroku-specific optimizations
node scripts/heroku-atlas-optimizer.js
```

3. **Build Script Issues:**

```
// package.json
{
  "scripts": {
    "heroku-postbuild": "cd client && npm install && npm run build"
  }
}
```

MongoDB Atlas Performance

Symptoms: - High latency connections - Timeout errors - Poor query performance

Diagnosis:

```
# Check Atlas performance
node scripts/heroku-atlas-optimizer.js
```

Optimizations:

1. **Region Proximity:** - Ensure Heroku and Atlas are in same region - Use `us-east-1` for both services

2. **Connection Pooling:**

```
const mongoOptions = {
  maxPoolSize: 10,
  minPoolSize: 5,
  maxIdleTimeMS: 30000,
  serverSelectionTimeoutMS: 5000,
  socketTimeoutMS: 45000
};
```

Translation and Internationalization

Translation Workflow Issues

Symptoms: - Missing translation keys - Translation workflow not triggering - JSON syntax errors

Validation:

```
# Test translation setup
./scripts/test-translation-setup.sh
```

Common Fixes:

1. **File Structure:**

```
# Ensure correct structure
client/
├── locales/
│   ├── en.json
│   ├── fr.json
│   ├── pt.json
│   └── es.json
└── src/
    └── i18n.js
```

2. **JSON Validation:**

```
# Validate JSON files
python3 -m json.tool client/locales/en.json
```

3. **GitHub Workflow:**

```
# .github/workflows/translate_app.yml
name: Auto-translate Application
on:
  push:
    paths:
      - 'client/locales/en.json'
```

Code Quality Issues

Cleanup and Maintenance

Symptoms: - Excessive console.log statements - Commented-out code - Unused files and dependencies

Automated Cleanup:

```
# Run comprehensive cleanup
./scripts/cleanup.sh

# Code-specific cleanup
./scripts/cleanup_script.sh
```

Manual Cleanup:

```
# Remove debugging code
find . -name "*.js" | xargs grep -l "console.log" | head -10

# Find commented code
find . -name "*.js" | xargs grep -l "^[[:space:]]*//.*TODO"
```

Security Issues

Symptoms: - Exposed credentials in repository - Security warnings - Vulnerable dependencies

Immediate Actions:

```
# Check for exposed secrets
git log --all --full-history -- .env*

# Security audit
npm audit

# Fix vulnerabilities
npm audit fix
```

Prevention:

```
# .gitignore
.env
.env.*
!.env.example
```

Performance Optimization

Monitoring and Validation

Real-time Monitoring:

```
# Monitor performance
node scripts/monitor-performance.js
```

Load Testing:

```
# Test endpoint performance
node scripts/test-optimized-endpoints.js
```

Performance Benchmarks:

```
# Comprehensive performance test
node scripts/test-performance.js
```

Expected Results: - K. pneumoniae: <2s load time - E. coli: <7s with pagination - E. coli (diarrheagenic): <3s load time - Payload reduction: 60-90%

Diagnostic Scripts Reference

Quick Diagnostics:

```
# Health check
curl http://localhost:8080/api/health

# Performance check
node scripts/real-performance-test.js

# Deployment readiness
node scripts/deployment-readiness.js
```

Comprehensive Analysis:

```
# Full performance analysis
node scripts/organism-performance-debugger.js

# Heroku/Atlas optimization
node scripts/heroku-atlas-optimizer.js

# Translation validation
./scripts/test-translation-setup.sh
```

Status Checks:

```
# Project status
./scripts/status-check.sh

# API endpoints test
./scripts/test-api.sh
```

Emergency Procedures

Critical Performance Issues

If users report 2+ minute load times:

1. **Immediate Response:**

```
# Switch to optimized endpoints
# Update frontend API calls from /api/ to /api/optimized/
```

2. **Verify Fix:**

```
node scripts/post-fix-validation.js
```

3. **Monitor Results:**

```
node scripts/monitor-performance.js
```

Server Down Emergency

If server is unresponsive:

1. **Start Minimal Server:**

```
node scripts/minimal-server.js
```

2. **Check Logs:**

```
heroku logs --tail
```

3. **Restart with Fixed Configuration:**

```
node scripts/server-fixed.js
```

Getting Help

When to Use Each Script:

- **Performance Issues**: `organism-performance-debugger.js`
- **Deployment Problems**: `deployment-readiness.js`
- **MongoDB Issues**: `test-fixie-connection.js`
- **Code Quality**: `cleanup_script.sh`
- **Translation Issues**: `test-translation-setup.sh`

Community Support: - GitHub Issues: https://github.com/amrnet/amrnet/issues - Discussions: https://github.com/amrnet/amrnet/discussions - Email: amrnetdashboard@gmail.com

Emergency Contacts: - Critical bugs: GitHub Issues with "urgent" label - Security issues: amrnetdashboard@gmail.com

2.5.7 Contributor Guide: Issues & PRs

Thank you for your interest in improving this project. This project is open-source under the GPL-3.0 license. and welcomes contributions in the form of bug reports, feature requests, and pull requests.

Here is a list of important resources for contributors:

- Source Code
- Documentation
- Issue Tracker
- *Code of Conduct*

How to report a bug

Report bugs on the Issue Tracker.

When filing an issue, make sure to answer these questions:

- Which operating system and Python version are you using?
- Which version of this project are you using?
- What did you do?
- What did you expect to see?

- What did you see instead?

The best way to get your bug fixed is to provide a test case, and/or steps to reproduce the issue.

How to request a feature

Features that improve `AMRnet` can be suggested on the Issue Tracker. It's a good idea to first submit the proposal as a feature request prior to submitting a pull request as this allows for the best coordination of efforts by preventing the duplication of work, and allows for feedback on your ideas.

2.5.8 Contributor Guide: Research & Development

Thank you for your interest in contributing to AMRnet! We welcome contributions from researchers, developers, and public health professionals worldwide. This project is open-source under the GPL-3.0 license and welcomes contributions in the form of bug reports, feature requests, and pull requests.

Here is a list of important resources for contributors:

- Source Code
- Documentation
- Issue Tracker
- Discussions
- *Code of Conduct*

Quick Start

Ready to contribute? Here's how to set up AMRnet for local development:

1. **Fork** the repository on GitHub
2. **Clone** your fork locally
3. **Create** a feature branch for your changes
4. **Make** your changes and test them
5. **Submit** a pull request

Types of Contributions

Bug Reports

Help us improve by reporting bugs using our Issue Tracker.

Before reporting a bug: - Search existing issues to avoid duplicates - Test with the latest version - Check if the issue occurs in different browsers

When filing a bug report, please include: - Which operating system and browser you're using - Which version of AMRnet you're using - Steps to reproduce the issue - What you expected to happen - What actually happened - Screenshots if applicable

Feature Requests

We welcome feature suggestions that improve AMRnet's functionality for AMR surveillance.

Good feature requests include: - Clear description of the problem being solved - Specific use cases from public health or research perspectives - Consideration of different user types (researchers, policymakers, etc.)

Documentation Improvements

Documentation improvements are always welcome, including: - Fixing typos or clarifying instructions - Adding examples or use cases - Translating content to other languages - Creating tutorials or guides

How to set up your development environment

Prerequisites

Before you begin, ensure you have:

- **Node.js** 18+ with npm
- **Python** 3.8+ (for data processing)
- **MongoDB** 6.0+ (local or Atlas)
- **Git** for version control

Initial Setup

```
# Clone your fork
git clone https://github.com/YOUR_USERNAME/amrnet.git
cd amrnet

# Install dependencies
npm install
cd client && npm install && cd ..

# Set up environment
cp .env.example .env
# Edit .env with your configuration

# Start development servers
npm run start:dev
```

The application will be available at `http://localhost:3000` with the API at `http://localhost:8080`.

Code Quality Standards

Linting and Formatting

AMRnet uses ESLint and Prettier to maintain code quality:

```
# Lint JavaScript/React code
cd client && npm run lint

# Format code with Prettier
npm run format

# Fix linting issues automatically
cd client && npm run lint:fix
```

Testing

Always include tests with your contributions:

```
# Run frontend tests
cd client && npm test

# Run tests with coverage
cd client && npm test -- --coverage

# Run backend tests (if available)
npm run test:backend
```

Git Hooks

Pre-commit hooks automatically run linting and formatting: - Configured via `.editorconfig` and `.prettierrc.json` - Ensures consistent code style across all contributions

Development Workflow

Branch Naming

Use descriptive branch names: - `feature/add-organism-filtering` - `bugfix/map-rendering-issue` - `docs/update-installation-guide`

Commit Messages

Follow conventional commit format: - `feat: add new organism filtering capability` - `fix: resolve map rendering issue on mobile` - `docs: update installation instructions`

Code Review Process

All contributions go through code review: 1. Create a pull request with clear description 2. Automated tests must pass 3. Code review by maintainers 4. Address feedback and update as needed 5. Merge once approved

Specific Contribution Areas

Frontend Development

Technologies: React 18, Material-UI, Redux, Recharts

Key areas for contribution: - New visualization components - Mobile responsiveness improvements - Accessibility enhancements - Performance optimizations

Backend Development

Technologies: Node.js, Express.js, MongoDB

Key areas for contribution: - API endpoint optimization - Database query improvements - Data validation and processing - Security enhancements

Data Processing

Technologies: Python, pandas, NumPy

Key areas for contribution: - New organism data parsers - Data quality validation - Statistical analysis functions - Export format support

Documentation

Areas needing help: - User guides and tutorials - API documentation - Developer onboarding - Multi-language translations

How to submit changes

Open a pull request to submit changes to this project.

Your pull request should: - Include a clear description of changes - Pass all automated tests - Include relevant tests for new functionality - Update documentation if needed - Follow the project's coding standards

Pull request template includes: - Description of changes - Type of change (bugfix, feature, docs, etc.) - Testing checklist - Screenshots for UI changes

Community Guidelines

- Be respectful and inclusive
- Provide constructive feedback
- Focus on the scientific and public health mission
- Help newcomers get started
- Follow our *Code of Conduct*

Getting Help

Need assistance? - Check existing Issues - Join our Discussions - Review the *Installation Guide* - Read the Developer Guide

For urgent issues: - Security vulnerabilities: See our Security Policy - Critical bugs: Use the Issue Tracker with "urgent" label

2.5.9 Code of Conduct

Our Pledge

We as members, contributors, and leaders pledge to make participation in our community a harassment-free experience for everyone, regardless of age, body size, visible or invisible disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, caste, color, religion, or sexual identity and orientation.

We pledge to act and interact in ways that contribute to an open, welcoming, diverse, inclusive, and healthy community.

Our Standards

Examples of behavior that contributes to a positive environment for our community include:

- Demonstrating empathy and kindness toward other people
- Being respectful of differing opinions, viewpoints, and experiences
- Giving and gracefully accepting constructive feedback

- Accepting responsibility and apologizing to those affected by our mistakes, and learning from the experience
- Focusing on what is best not just for us as individuals, but for the overall community

Examples of unacceptable behavior include:

- The use of sexualized language or imagery, and sexual attention or advances of any kind
- Trolling, insulting or derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or email address, without their explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

Enforcement Responsibilities

Community leaders are responsible for clarifying and enforcing our standards of acceptable behavior and will take appropriate and fair corrective action in response to any behavior that they deem inappropriate, threatening, offensive, or harmful.

Community leaders have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, and will communicate reasons for moderation decisions when appropriate.

Scope

This Code of Conduct applies within all community spaces, and also applies when an individual is officially representing the community in public spaces. Examples of representing our community include using an official e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event.

Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported to the community leaders responsible for enforcement at amrnetdashboard@gmail.com. All complaints will be reviewed and investigated promptly and fairly.

All community leaders are obligated to respect the privacy and security of the reporter of any incident.

Enforcement Guidelines

Community leaders will follow these Community Impact Guidelines in determining the consequences for any action they deem in violation of this Code of Conduct:

1. Correction

Community Impact: Use of inappropriate language or other behavior deemed unprofessional or unwelcome in the community.

Consequence: A private, written warning from community leaders, providing clarity around the nature of the violation and an explanation of why the behavior was inappropriate. A public apology may be requested.

2. Warning

Community Impact: A violation through a single incident or series of actions.

Consequence: A warning with consequences for continued behavior. No interaction with the people involved, including unsolicited interaction with those enforcing the Code of Conduct, for a specified period of time. This includes avoiding interactions in community spaces as well as external channels like social media. Violating these terms may lead to a temporary or permanent ban.

3. Temporary Ban

Community Impact: A serious violation of community standards, including sustained inappropriate behavior.

Consequence: A temporary ban from any sort of interaction or public communication with the community for a specified period of time. No public or private interaction with the people involved, including unsolicited interaction with those enforcing the Code of Conduct, is allowed during this period. Violating these terms may lead to a permanent ban.

4. Permanent Ban

Community Impact: Demonstrating a pattern of violation of community standards, including sustained inappropriate behavior, harassment of an individual, or aggression toward or disparagement of classes of individuals.

Consequence: A permanent ban from any sort of public interaction within the community.

Attribution

This Code of Conduct is adapted from the Contributor Covenant, version 2.1, available at https://www.contributor-covenant.org/version/2/1/code_of_conduct.html.

Community Impact Guidelines were inspired by Mozilla's code of conduct enforcement ladder.

For answers to common questions about this code of conduct, see the FAQ at https://www.contributor-covenant.org/faq. Translations are available at https://www.contributor-covenant.org/translations.

2.5.10 License

```
                    GNU GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

 Copyright (C) 2007 Free Software Foundation, Inc. <https://fsf.org/>
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

                            Preamble

  The GNU General Public License is a free, copyleft license for
software and other kinds of works.

  The licenses for most software and other practical works are designed
to take away your freedom to share and change the works.  By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users.  We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors.  You can apply it to
your programs, too.

  When we speak of free software, we are referring to freedom, not
price.  Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.
```

(continues on next page)

(continued from previous page)

```
  To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights.  Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

  For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received.  You must make sure that they, too, receive
or can get the source code.  And you must show them these terms so they
know their rights.

  Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

  For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software.  For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

  Some devices are designed to deny users access to install or run
modified versions of the software inside them, although the manufacturer
can do so.  This is fundamentally incompatible with the aim of
protecting users' freedom to change the software.  The systematic
pattern of such abuse occurs in the area of products for individuals to
use, which is precisely where it is most unacceptable.  Therefore, we
have designed this version of the GPL to prohibit the practice for those
products.  If such problems arise substantially in other domains, we
stand ready to extend this provision to those domains in future versions
of the GPL, as needed to protect the freedom of users.

  Finally, every program is threatened constantly by software patents.
States should not allow patents to restrict development and use of
software on general-purpose computers, but in those that do, we wish to
avoid the special danger that patents applied to a free program could
make it effectively proprietary.  To prevent this, the GPL assures that
patents cannot be used to render the program non-free.

  The precise terms and conditions for copying, distribution and
modification follow.

                       TERMS AND CONDITIONS

  0. Definitions.

  "This License" refers to version 3 of the GNU General Public License.

  "Copyright" also means copyright-like laws that apply to other kinds of
works, such as semiconductor masks.
```

(continues on next page)

(continued from previous page)

```
  "The Program" refers to any copyrightable work licensed under this
License.  Each licensee is addressed as "you".  "Licensees" and
"recipients" may be individuals or organizations.

  To "modify" a work means to copy from or adapt all or part of the work
in a fashion requiring copyright permission, other than the making of an
exact copy.  The resulting work is called a "modified version" of the
earlier work or a work "based on" the earlier work.

  A "covered work" means either the unmodified Program or a work based
on the Program.

  To "propagate" a work means to do anything with it that, without
permission, would make you directly or secondarily liable for
infringement under applicable copyright law, except executing it on a
computer or modifying a private copy.  Propagation includes copying,
distribution (with or without modification), making available to the
public, and in some countries other activities as well.

  To "convey" a work means any kind of propagation that enables other
parties to make or receive copies.  Mere interaction with a user through
a computer network, with no transfer of a copy, is not conveying.

  An interactive user interface displays "Appropriate Legal Notices"
to the extent that it includes a convenient and prominently visible
feature that (1) displays an appropriate copyright notice, and (2)
tells the user that there is no warranty for the work (except to the
extent that warranties are provided), that licensees may convey the
work under this License, and how to view a copy of this License.  If
the interface presents a list of user commands or options, such as a
menu, a prominent item in the list meets this criterion.

  1. Source Code.

  The "source code" for a work means the preferred form of the work
for making modifications to it.  "Object code" means any non-source
form of a work.

  A "Standard Interface" means an interface that either is an official
standard defined by a recognized standards body, or, in the case of
interfaces specified for a particular programming language, one that
is widely used among developers working in that language.

  The "System Libraries" of an executable work include anything, other
than the work as a whole, that (a) is included in the normal form of
packaging a Major Component, but which is not part of that Major
Component, and (b) serves only to enable use of the work with that
Major Component, or to implement a Standard Interface for which an
implementation is available to the public in source code form.  A
"Major Component", in this context, means a major essential component
(kernel, window system, and so on) of the specific operating system
(if any) on which the executable work runs, or a compiler used to
```

(continues on next page)

(continued from previous page)

```
produce the work, or an object code interpreter used to run it.

  The "Corresponding Source" for a work in object code form means all
the source code needed to generate, install, and (for an executable
work) run the object code and to modify the work, including scripts to
control those activities.  However, it does not include the work's
System Libraries, or general-purpose tools or generally available free
programs which are used unmodified in performing those activities but
which are not part of the work.  For example, Corresponding Source
includes interface definition files associated with source files for
the work, and the source code for shared libraries and dynamically
linked subprograms that the work is specifically designed to require,
such as by intimate data communication or control flow between those
subprograms and other parts of the work.

  The Corresponding Source need not include anything that users
can regenerate automatically from other parts of the Corresponding
Source.

  The Corresponding Source for a work in source code form is that
same work.

  2. Basic Permissions.

  All rights granted under this License are granted for the term of
copyright on the Program, and are irrevocable provided the stated
conditions are met.  This License explicitly affirms your unlimited
permission to run the unmodified Program.  The output from running a
covered work is covered by this License only if the output, given its
content, constitutes a covered work.  This License acknowledges your
rights of fair use or other equivalent, as provided by copyright law.

  You may make, run and propagate covered works that you do not
convey, without conditions so long as your license otherwise remains
in force.  You may convey covered works to others for the sole purpose
of having them make modifications exclusively for you, or provide you
with facilities for running those works, provided that you comply with
the terms of this License in conveying all material for which you do
not control copyright.  Those thus making or running the covered works
for you must do so exclusively on your behalf, under your direction
and control, on terms that prohibit them from making any copies of
your copyrighted material outside their relationship with you.

  Conveying under any other circumstances is permitted solely under
the conditions stated below.  Sublicensing is not allowed; section 10
makes it unnecessary.

  3. Protecting Users' Legal Rights From Anti-Circumvention Law.

  No covered work shall be deemed part of an effective technological
measure under any applicable law fulfilling obligations under article
11 of the WIPO copyright treaty adopted on 20 December 1996, or
```

(continues on next page)

(continued from previous page)

```
similar laws prohibiting or restricting circumvention of such
measures.

  When you convey a covered work, you waive any legal power to forbid
circumvention of technological measures to the extent such circumvention
is effected by exercising rights under this License with respect to
the covered work, and you disclaim any intention to limit operation or
modification of the work as a means of enforcing, against the work's
users, your or third parties' legal rights to forbid circumvention of
technological measures.

  4. Conveying Verbatim Copies.

  You may convey verbatim copies of the Program's source code as you
receive it, in any medium, provided that you conspicuously and
appropriately publish on each copy an appropriate copyright notice;
keep intact all notices stating that this License and any
non-permissive terms added in accord with section 7 apply to the code;
keep intact all notices of the absence of any warranty; and give all
recipients a copy of this License along with the Program.

  You may charge any price or no price for each copy that you convey,
and you may offer support or warranty protection for a fee.

  5. Conveying Modified Source Versions.

  You may convey a work based on the Program, or the modifications to
produce it from the Program, in the form of source code under the
terms of section 4, provided that you also meet all of these conditions:

    a) The work must carry prominent notices stating that you modified
    it, and giving a relevant date.

    b) The work must carry prominent notices stating that it is
    released under this License and any conditions added under section
    7.  This requirement modifies the requirement in section 4 to
    "keep intact all notices".

    c) You must license the entire work, as a whole, under this
    License to anyone who comes into possession of a copy.  This
    License will therefore apply, along with any applicable section 7
    additional terms, to the whole of the work, and all its parts,
    regardless of how they are packaged.  This License gives no
    permission to license the work in any other way, but it does not
    invalidate such permission if you have separately received it.

    d) If the work has interactive user interfaces, each must display
    Appropriate Legal Notices; however, if the Program has interactive
    interfaces that do not display Appropriate Legal Notices, your
    work need not make them do so.

  A compilation of a covered work with other separate and independent
```

(continues on next page)

(continued from previous page)

```
works, which are not by their nature extensions of the covered work,
and which are not combined with it such as to form a larger program,
in or on a volume of a storage or distribution medium, is called an
"aggregate" if the compilation and its resulting copyright are not
used to limit the access or legal rights of the compilation's users
beyond what the individual works permit.  Inclusion of a covered work
in an aggregate does not cause this License to apply to the other
parts of the aggregate.

  6. Conveying Non-Source Forms.

  You may convey a covered work in object code form under the terms
of sections 4 and 5, provided that you also convey the
machine-readable Corresponding Source under the terms of this License,
in one of these ways:

    a) Convey the object code in, or embodied in, a physical product
    (including a physical distribution medium), accompanied by the
    Corresponding Source fixed on a durable physical medium
    customarily used for software interchange.

    b) Convey the object code in, or embodied in, a physical product
    (including a physical distribution medium), accompanied by a
    written offer, valid for at least three years and valid for as
    long as you offer spare parts or customer support for that product
    model, to give anyone who possesses the object code either (1) a
    copy of the Corresponding Source for all the software in the
    product that is covered by this License, on a durable physical
    medium customarily used for software interchange, for a price no
    more than your reasonable cost of physically performing this
    conveying of source, or (2) access to copy the
    Corresponding Source from a network server at no charge.

    c) Convey individual copies of the object code with a copy of the
    written offer to provide the Corresponding Source.  This
    alternative is allowed only occasionally and noncommercially, and
    only if you received the object code with such an offer, in accord
    with subsection 6b.

    d) Convey the object code by offering access from a designated
    place (gratis or for a charge), and offer equivalent access to the
    Corresponding Source in the same way through the same place at no
    further charge.  You need not require recipients to copy the
    Corresponding Source along with the object code.  If the place to
    copy the object code is a network server, the Corresponding Source
    may be on a different server (operated by you or a third party)
    that supports equivalent copying facilities, provided you maintain
    clear directions next to the object code saying where to find the
    Corresponding Source.  Regardless of what server hosts the
    Corresponding Source, you remain obligated to ensure that it is
    available for as long as needed to satisfy these requirements.
```

(continues on next page)

(continued from previous page)

```
    e) Convey the object code using peer-to-peer transmission, provided
    you inform other peers where the object code and Corresponding
    Source of the work are being offered to the general public at no
    charge under subsection 6d.

  A separable portion of the object code, whose source code is excluded
from the Corresponding Source as a System Library, need not be
included in conveying the object code work.

  A "User Product" is either (1) a "consumer product", which means any
tangible personal property which is normally used for personal, family,
or household purposes, or (2) anything designed or sold for incorporation
into a dwelling.  In determining whether a product is a consumer product,
doubtful cases shall be resolved in favor of coverage.  For a particular
product received by a particular user, "normally used" refers to a
typical or common use of that class of product, regardless of the status
of the particular user or of the way in which the particular user
actually uses, or expects or is expected to use, the product.  A product
is a consumer product regardless of whether the product has substantial
commercial, industrial or non-consumer uses, unless such uses represent
the only significant mode of use of the product.

  "Installation Information" for a User Product means any methods,
procedures, authorization keys, or other information required to install
and execute modified versions of a covered work in that User Product from
a modified version of its Corresponding Source.  The information must
suffice to ensure that the continued functioning of the modified object
code is in no case prevented or interfered with solely because
modification has been made.

  If you convey an object code work under this section in, or with, or
specifically for use in, a User Product, and the conveying occurs as
part of a transaction in which the right of possession and use of the
User Product is transferred to the recipient in perpetuity or for a
fixed term (regardless of how the transaction is characterized), the
Corresponding Source conveyed under this section must be accompanied
by the Installation Information.  But this requirement does not apply
if neither you nor any third party retains the ability to install
modified object code on the User Product (for example, the work has
been installed in ROM).

  The requirement to provide Installation Information does not include a
requirement to continue to provide support service, warranty, or updates
for a work that has been modified or installed by the recipient, or for
the User Product in which it has been modified or installed.  Access to a
network may be denied when the modification itself materially and
adversely affects the operation of the network or violates the rules and
protocols for communication across the network.

  Corresponding Source conveyed, and Installation Information provided,
in accord with this section must be in a format that is publicly
documented (and with an implementation available to the public in
```

(continues on next page)

(continued from previous page)

```
source code form), and must require no special password or key for
unpacking, reading or copying.

  7. Additional Terms.

  "Additional permissions" are terms that supplement the terms of this
License by making exceptions from one or more of its conditions.
Additional permissions that are applicable to the entire Program shall
be treated as though they were included in this License, to the extent
that they are valid under applicable law.  If additional permissions
apply only to part of the Program, that part may be used separately
under those permissions, but the entire Program remains governed by
this License without regard to the additional permissions.

  When you convey a copy of a covered work, you may at your option
remove any additional permissions from that copy, or from any part of
it.  (Additional permissions may be written to require their own
removal in certain cases when you modify the work.)  You may place
additional permissions on material, added by you to a covered work,
for which you have or can give appropriate copyright permission.

  Notwithstanding any other provision of this License, for material you
add to a covered work, you may (if authorized by the copyright holders of
that material) supplement the terms of this License with terms:

    a) Disclaiming warranty or limiting liability differently from the
    terms of sections 15 and 16 of this License; or

    b) Requiring preservation of specified reasonable legal notices or
    author attributions in that material or in the Appropriate Legal
    Notices displayed by works containing it; or

    c) Prohibiting misrepresentation of the origin of that material, or
    requiring that modified versions of such material be marked in
    reasonable ways as different from the original version; or

    d) Limiting the use for publicity purposes of names of licensors or
    authors of the material; or

    e) Declining to grant rights under trademark law for use of some
    trade names, trademarks, or service marks; or

    f) Requiring indemnification of licensors and authors of that
    material by anyone who conveys the material (or modified versions of
    it) with contractual assumptions of liability to the recipient, for
    any liability that these contractual assumptions directly impose on
    those licensors and authors.

  All other non-permissive additional terms are considered "further
restrictions" within the meaning of section 10.  If the Program as you
received it, or any part of it, contains a notice stating that it is
governed by this License along with a term that is a further
```

(continues on next page)

(continued from previous page)

```
restriction, you may remove that term.  If a license document contains
a further restriction but permits relicensing or conveying under this
License, you may add to a covered work material governed by the terms
of that license document, provided that the further restriction does
not survive such relicensing or conveying.

  If you add terms to a covered work in accord with this section, you
must place, in the relevant source files, a statement of the
additional terms that apply to those files, or a notice indicating
where to find the applicable terms.

  Additional terms, permissive or non-permissive, may be stated in the
form of a separately written license, or stated as exceptions;
the above requirements apply either way.

  8. Termination.

  You may not propagate or modify a covered work except as expressly
provided under this License.  Any attempt otherwise to propagate or
modify it is void, and will automatically terminate your rights under
this License (including any patent licenses granted under the third
paragraph of section 11).

  However, if you cease all violation of this License, then your
license from a particular copyright holder is reinstated (a)
provisionally, unless and until the copyright holder explicitly and
finally terminates your license, and (b) permanently, if the copyright
holder fails to notify you of the violation by some reasonable means
prior to 60 days after the cessation.

  Moreover, your license from a particular copyright holder is
reinstated permanently if the copyright holder notifies you of the
violation by some reasonable means, this is the first time you have
received notice of violation of this License (for any work) from that
copyright holder, and you cure the violation prior to 30 days after
your receipt of the notice.

  Termination of your rights under this section does not terminate the
licenses of parties who have received copies or rights from you under
this License.  If your rights have been terminated and not permanently
reinstated, you do not qualify to receive new licenses for the same
material under section 10.

  9. Acceptance Not Required for Having Copies.

  You are not required to accept this License in order to receive or
run a copy of the Program.  Ancillary propagation of a covered work
occurring solely as a consequence of using peer-to-peer transmission
to receive a copy likewise does not require acceptance.  However,
nothing other than this License grants you permission to propagate or
modify any covered work.  These actions infringe copyright if you do
not accept this License.  Therefore, by modifying or propagating a
```

(continues on next page)

(continued from previous page)

```
covered work, you indicate your acceptance of this License to do so.

  10. Automatic Licensing of Downstream Recipients.

  Each time you convey a covered work, the recipient automatically
receives a license from the original licensors, to run, modify and
propagate that work, subject to this License.  You are not responsible
for enforcing compliance by third parties with this License.

  An "entity transaction" is a transaction transferring control of an
organization, or substantially all assets of one, or subdividing an
organization, or merging organizations.  If propagation of a covered
work results from an entity transaction, each party to that
transaction who receives a copy of the work also receives whatever
licenses to the work the party's predecessor in interest had or could
give under the previous paragraph, plus a right to possession of the
Corresponding Source of the work from the predecessor in interest, if
the predecessor has it or can get it with reasonable efforts.

  You may not impose any further restrictions on the exercise of the
rights granted or affirmed under this License.  For example, you may
not impose a license fee, royalty, or other charge for exercise of
rights granted under this License, and you may not initiate litigation
(including a cross-claim or counterclaim in a lawsuit) alleging that
any patent claim is infringed by making, using, selling, offering for
sale, or importing the Program or any portion of it.

  11. Patents.

  A "contributor" is a copyright holder who authorizes use under this
License of the Program or a work on which the Program is based.  The
work thus licensed is called the contributor's "contributor version".

  A contributor's "essential patent claims" are all patent claims
owned or controlled by the contributor, whether already acquired or
hereafter acquired, that would be infringed by some manner, permitted
by this License, of making, using, or selling its contributor version,
but do not include claims that would be infringed only as a
consequence of further modification of the contributor version.  For
purposes of this definition, "control" includes the right to grant
patent sublicenses in a manner consistent with the requirements of
this License.

  Each contributor grants you a non-exclusive, worldwide, royalty-free
patent license under the contributor's essential patent claims, to
make, use, sell, offer for sale, import and otherwise run, modify and
propagate the contents of its contributor version.

  In the following three paragraphs, a "patent license" is any express
agreement or commitment, however denominated, not to enforce a patent
(such as an express permission to practice a patent or covenant not to
sue for patent infringement).  To "grant" such a patent license to a
```

(continues on next page)

(continued from previous page)

```
party means to make such an agreement or commitment not to enforce a
patent against the party.

  If you convey a covered work, knowingly relying on a patent license,
and the Corresponding Source of the work is not available for anyone
to copy, free of charge and under the terms of this License, through a
publicly available network server or other readily accessible means,
then you must either (1) cause the Corresponding Source to be so
available, or (2) arrange to deprive yourself of the benefit of the
patent license for this particular work, or (3) arrange, in a manner
consistent with the requirements of this License, to extend the patent
license to downstream recipients.  "Knowingly relying" means you have
actual knowledge that, but for the patent license, your conveying the
covered work in a country, or your recipient's use of the covered work
in a country, would infringe one or more identifiable patents in that
country that you have reason to believe are valid.

  If, pursuant to or in connection with a single transaction or
arrangement, you convey, or propagate by procuring conveyance of, a
covered work, and grant a patent license to some of the parties
receiving the covered work authorizing them to use, propagate, modify
or convey a specific copy of the covered work, then the patent license
you grant is automatically extended to all recipients of the covered
work and works based on it.

  A patent license is "discriminatory" if it does not include within
the scope of its coverage, prohibits the exercise of, or is
conditioned on the non-exercise of one or more of the rights that are
specifically granted under this License.  You may not convey a covered
work if you are a party to an arrangement with a third party that is
in the business of distributing software, under which you make payment
to the third party based on the extent of your activity of conveying
the work, and under which the third party grants, to any of the
parties who would receive the covered work from you, a discriminatory
patent license (a) in connection with copies of the covered work
conveyed by you (or copies made from those copies), or (b) primarily
for and in connection with specific products or compilations that
contain the covered work, unless you entered into that arrangement,
or that patent license was granted, prior to 28 March 2007.

  Nothing in this License shall be construed as excluding or limiting
any implied license or other defenses to infringement that may
otherwise be available to you under applicable patent law.

  12. No Surrender of Others' Freedom.

  If conditions are imposed on you (whether by court order, agreement or
otherwise) that contradict the conditions of this License, they do not
excuse you from the conditions of this License.  If you cannot convey a
covered work so as to satisfy simultaneously your obligations under this
License and any other pertinent obligations, then as a consequence you may
not convey it at all.  For example, if you agree to terms that obligate you
```

(continues on next page)

(continued from previous page)

```
to collect a royalty for further conveying from those to whom you convey
the Program, the only way you could satisfy both those terms and this
License would be to refrain entirely from conveying the Program.

  13. Use with the GNU Affero General Public License.

  Notwithstanding any other provision of this License, you have
permission to link or combine any covered work with a work licensed
under version 3 of the GNU Affero General Public License into a single
combined work, and to convey the resulting work.  The terms of this
License will continue to apply to the part which is the covered work,
but the special requirements of the GNU Affero General Public License,
section 13, concerning interaction through a network will apply to the
combination as such.

  14. Revised Versions of this License.

  The Free Software Foundation may publish revised and/or new versions of
the GNU General Public License from time to time.  Such new versions will
be similar in spirit to the present version, but may differ in detail to
address new problems or concerns.

  Each version is given a distinguishing version number.  If the
Program specifies that a certain numbered version of the GNU General
Public License "or any later version" applies to it, you have the
option of following the terms and conditions either of that numbered
version or of any later version published by the Free Software
Foundation.  If the Program does not specify a version number of the
GNU General Public License, you may choose any version ever published
by the Free Software Foundation.

  If the Program specifies that a proxy can decide which future
versions of the GNU General Public License can be used, that proxy's
public statement of acceptance of a version permanently authorizes you
to choose that version for the Program.

  Later license versions may give you additional or different
permissions.  However, no additional obligations are imposed on any
author or copyright holder as a result of your choosing to follow a
later version.

  15. Disclaimer of Warranty.

  THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY
APPLICABLE LAW.  EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT
HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY
OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE.  THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM
IS WITH YOU.  SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF
ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
```

(continues on next page)

(continued from previous page)

```
  16. Limitation of Liability.

  IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING
WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS
THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY
GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE
USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF
DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD
PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS),
EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF
SUCH DAMAGES.

  17. Interpretation of Sections 15 and 16.

  If the disclaimer of warranty and limitation of liability provided
above cannot be given local legal effect according to their terms,
reviewing courts shall apply local law that most closely approximates
an absolute waiver of all civil liability in connection with the
Program, unless a warranty or assumption of liability accompanies a
copy of the Program in return for a fee.

                     END OF TERMS AND CONDITIONS

            How to Apply These Terms to Your New Programs

  If you develop a new program, and you want it to be of the greatest
possible use to the public, the best way to achieve this is to make it
free software which everyone can redistribute and change under these terms.

  To do so, attach the following notices to the program.  It is safest
to attach them to the start of each source file to most effectively
state the exclusion of warranty; and each file should have at least
the "copyright" line and a pointer to where the full notice is found.

    <one line to give the program's name and a brief idea of what it does.>
    Copyright (C) <year>  <name of author>

    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program.  If not, see <https://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.
```

(continues on next page)

(continued from previous page)

```
  If the program does terminal interaction, make it output a short
notice like this when it starts in an interactive mode:

    <program>  Copyright (C) <year>  <name of author>
    This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License.  Of course, your program's commands
might be different; for a GUI interface, you would use an "about box".

  You should also get your employer (if you work as a programmer) or school,
if any, to sign a "copyright disclaimer" for the program, if necessary.
For more information on this, and how to apply and follow the GNU GPL, see
<https://www.gnu.org/licenses/>.

  The GNU General Public License does not permit incorporating your program
into proprietary programs.  If your program is a subroutine library, you
may consider it more useful to permit linking proprietary applications with
the library.  If this is what you want to do, use the GNU Lesser General
Public License instead of this License.  But first, please read
<https://www.gnu.org/licenses/why-not-lgpl.html>.
```

These instructions will get you a copy of `AMRnet` up and running on your machine for development and testing purposes. AMRnet is a full-stack web application built with Node.js, React, and MongoDB.

2.5.11 Prerequisites

Before installing AMRnet, make sure you have the following software installed:

- **Node.js**: v18.20.4 or higher (specified in `.nvmrc`)
- **npm**: Latest version (comes with Node.js)
- **Python**: v3.8+ (for data processing components)
- **Git**: Latest version for version control
- **MongoDB**: v6.0+ (local installation; production runs MongoDB on the application EC2 host and exposes it via SSH tunnel)

You can check your installed versions with:

```
node --version
npm --version
python --version
git --version
```

2.5.12 Quick Start

Follow these steps to get AMRnet running locally:

1. Clone the Repository

```
git clone https://github.com/amrnet/amrnet.git
cd amrnet
```

2. Install Dependencies

```
# Install backend dependencies
npm install

# Install frontend dependencies
cd client && npm install && cd ..
```

3. Environment Configuration

```
# Create environment file from template
cp .env.example .env

# Edit .env file with your configuration
# Add your MongoDB connection string and other settings
```

4. Start Development Servers

```
# Start both backend and frontend in development mode
npm run start:dev

# Or start them individually:
npm run start:backend  # Backend only (port 8080)
npm run client         # Frontend only (port 3000)
```

5. Access the Application

Open your browser and navigate to `http://localhost:3000` to see the AMRnet dashboard.

2.5.13 Development Setup

For detailed development setup, including code quality tools and best practices:

Node.js Version Management

AMRnet uses Node.js v18.20.4. If you use nvm (Node Version Manager):

```
# Use the project's specified Node.js version
nvm use

# Or install the specific version if not available
nvm install v18.20.4
nvm use v18.20.4
```

Python Dependencies

For data processing components and documentation building:

```
# Install Python dependencies
pip install -r requirements.txt
```

(continues on next page)

(continued from previous page)

```
# Install documentation dependencies
pip install -r docs/requirements.txt
```

Environment Variables

Configure your `.env` file with the following variables:

```
# Application settings
NODE_ENV=development
PORT=8080

# Database configuration
MONGODB_URI=mongodb://localhost:27017/amrnet
# Production hosts MongoDB on the application EC2 instance (bound to
# 127.0.0.1). Use ssh -L 27018:127.0.0.1:27017 to reach it from Compass.
```

2.5.14 Production Build

To build AMRnet for production deployment:

```
# Build the client application
npm run build

# Start the production server
npm start
```

The built application will be served from the `client/build` directory.

2.5.15 Docker Installation

AMRnet can also be run using Docker:

```
# Build the Docker image
docker build -t amrnet .

# Run the container
docker run -p 8080:8080 -e MONGODB_URI=your_mongodb_uri amrnet
```

Make sure to replace `your_mongodb_uri` with your actual MongoDB connection string.

2.5.16 Troubleshooting

Common Installation Issues:

1. **Node version mismatch**: Use `nvm use` to switch to the correct version
2. **Package conflicts**: Delete `node_modules` and run `npm install` again
3. **Port conflicts**: Make sure ports 3000 and 8080 are available
4. **MongoDB connection**: Verify the MongoDB service is running locally, or that the SSH tunnel to the EC2 host is open

Getting Help:

- Check the Issue Tracker

- Review the Development Guide
- Join our Discussions