# TSAMS GitHub Technical Guide

## Repository Structure

The TSAMS ecosystem is organized into multiple specialized repositories:

### 1. tsams-core

Repository: github.com/ctibedoJ/tsams-core

Core mathematical structures and fundamental algorithms:

tsams-core/  
├── tsams\_core/  
│ ├── cyclotomic/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── cyclotomic\_field.py  
│ │ └── cyclotomic\_utils.py  
│ ├── mobius/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── mobius\_transformation.py  
│ │ └── root\_structures.py  
│ ├── state\_space/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── nodal\_structure.py  
│ │ └── state\_transitions.py  
│ ├── braid/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── hair\_braid\_dynamics.py  
│ │ └── braid\_operations.py  
│ ├── hyperbolic/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── hyperbolic\_priming.py  
│ │ └── optimization.py  
│ └── visualization/  
│ ├── \_\_init\_\_.py  
│ ├── field\_visualizer.py  
│ └── transformation\_plotter.py  
├── examples/  
│ ├── cyclotomic\_examples.py  
│ ├── mobius\_examples.py  
│ └── state\_space\_examples.py  
├── tests/  
│ ├── test\_cyclotomic.py  
│ ├── test\_mobius.py  
│ └── test\_state\_space.py  
├── docs/  
│ ├── api\_reference.md  
│ └── getting\_started.md  
├── setup.py  
├── requirements.txt  
├── README.md  
└── LICENSE

tsams-core/  
├── tsams\_core/  
│ ├── cyclotomic/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── cyclotomic\_field.py  
│ │ └── cyclotomic\_utils.py  
│ ├── mobius/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── mobius\_transformation.py  
│ │ └── root\_structures.py  
│ ├── state\_space/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── nodal\_structure.py  
│ │ └── state\_transitions.py  
│ ├── braid/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── hair\_braid\_dynamics.py  
│ │ └── braid\_operations.py  
│ ├── hyperbolic/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── hyperbolic\_priming.py  
│ │ └── optimization.py  
│ └── visualization/  
│ ├── \_\_init\_\_.py  
│ ├── field\_visualizer.py  
│ └── transformation\_plotter.py  
├── examples/  
│ ├── cyclotomic\_examples.py  
│ ├── mobius\_examples.py  
│ └── state\_space\_examples.py  
├── tests/  
│ ├── test\_cyclotomic.py  
│ ├── test\_mobius.py  
│ └── test\_state\_space.py  
├── docs/  
│ ├── api\_reference.md  
│ └── getting\_started.md  
├── setup.py  
├── requirements.txt  
├── README.md  
└── LICENSE

### 2. tsams-classical

Repository: github.com/ctibedoJ/tsams-classical

Classical mathematical implementations:

tsams-classical/  
├── tsams\_classical/  
│ ├── dedekind/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── dedekind\_cut.py  
│ │ └── continuity.py  
│ ├── prime/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── prime\_distribution.py  
│ │ └── prime\_utils.py  
│ ├── braid/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── braid\_group.py  
│ │ ├── braid\_word.py  
│ │ ├── braid\_invariants.py  
│ │ └── braid\_operations.py  
│ └── septimal/  
│ ├── \_\_init\_\_.py  
│ ├── septimal\_structures.py  
│ ├── septimal\_lattice.py  
│ └── septimal\_operations.py  
├── examples/  
│ ├── dedekind\_examples.py  
│ ├── prime\_examples.py  
│ └── braid\_examples.py  
├── tests/  
│ ├── test\_dedekind.py  
│ ├── test\_prime.py  
│ └── test\_braid.py  
├── docs/  
│ ├── api\_reference.md  
│ └── mathematical\_foundations.md  
├── setup.py  
├── requirements.txt  
├── README.md  
└── LICENSE

tsams-classical/  
├── tsams\_classical/  
│ ├── dedekind/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── dedekind\_cut.py  
│ │ └── continuity.py  
│ ├── prime/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── prime\_distribution.py  
│ │ └── prime\_utils.py  
│ ├── braid/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── braid\_group.py  
│ │ ├── braid\_word.py  
│ │ ├── braid\_invariants.py  
│ │ └── braid\_operations.py  
│ └── septimal/  
│ ├── \_\_init\_\_.py  
│ ├── septimal\_structures.py  
│ ├── septimal\_lattice.py  
│ └── septimal\_operations.py  
├── examples/  
│ ├── dedekind\_examples.py  
│ ├── prime\_examples.py  
│ └── braid\_examples.py  
├── tests/  
│ ├── test\_dedekind.py  
│ ├── test\_prime.py  
│ └── test\_braid.py  
├── docs/  
│ ├── api\_reference.md  
│ └── mathematical\_foundations.md  
├── setup.py  
├── requirements.txt  
├── README.md  
└── LICENSE

### 3. tsams-cryptography

Repository: github.com/ctibedoJ/tsams-cryptography

Cryptographic applications:

tsams-cryptography/  
├── tsams\_cryptography/  
│ ├── ecdlp/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── ecdlp\_solver.py  
│ │ └── optimization.py  
│ ├── lattice/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── lattice\_encryption.py  
│ │ └── lattice\_utils.py  
│ ├── hash/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── hash\_signatures.py  
│ │ └── merkle\_tree.py  
│ ├── code/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── code\_encryption.py  
│ │ └── error\_correction.py  
│ └── utils/  
│ ├── \_\_init\_\_.py  
│ ├── elliptic\_curves.py  
│ └── finite\_fields.py  
├── examples/  
│ ├── ecdlp\_examples.py  
│ ├── lattice\_examples.py  
│ └── hash\_examples.py  
├── tests/  
│ ├── test\_ecdlp.py  
│ ├── test\_lattice.py  
│ └── test\_hash.py  
├── docs/  
│ ├── cryptography\_guide.md  
│ └── security\_analysis.md  
├── setup.py  
├── requirements.txt  
├── README.md  
└── LICENSE

tsams-cryptography/  
├── tsams\_cryptography/  
│ ├── ecdlp/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── ecdlp\_solver.py  
│ │ └── optimization.py  
│ ├── lattice/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── lattice\_encryption.py  
│ │ └── lattice\_utils.py  
│ ├── hash/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── hash\_signatures.py  
│ │ └── merkle\_tree.py  
│ ├── code/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── code\_encryption.py  
│ │ └── error\_correction.py  
│ └── utils/  
│ ├── \_\_init\_\_.py  
│ ├── elliptic\_curves.py  
│ └── finite\_fields.py  
├── examples/  
│ ├── ecdlp\_examples.py  
│ ├── lattice\_examples.py  
│ └── hash\_examples.py  
├── tests/  
│ ├── test\_ecdlp.py  
│ ├── test\_lattice.py  
│ └── test\_hash.py  
├── docs/  
│ ├── cryptography\_guide.md  
│ └── security\_analysis.md  
├── setup.py  
├── requirements.txt  
├── README.md  
└── LICENSE

### 4. tsams-chemistry

Repository: github.com/ctibedoJ/tsams-chemistry

Chemical applications:

tsams-chemistry/  
├── tsams\_chemistry/  
│ ├── molecular/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── molecular\_modeling.py  
│ │ └── energy\_minimization.py  
│ ├── quantum/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── quantum\_chemistry.py  
│ │ └── orbital\_calculations.py  
│ ├── reaction/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── reaction\_pathways.py  
│ │ └── transition\_states.py  
│ └── visualization/  
│ ├── \_\_init\_\_.py  
│ ├── molecular\_visualizer.py  
│ └── reaction\_plotter.py  
├── examples/  
│ ├── molecular\_examples.py  
│ ├── quantum\_examples.py  
│ └── reaction\_examples.py  
├── tests/  
│ ├── test\_molecular.py  
│ ├── test\_quantum.py  
│ └── test\_reaction.py  
├── docs/  
│ ├── chemistry\_guide.md  
│ └── application\_examples.md  
├── setup.py  
├── requirements.txt  
├── README.md  
└── LICENSE

tsams-chemistry/  
├── tsams\_chemistry/  
│ ├── molecular/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── molecular\_modeling.py  
│ │ └── energy\_minimization.py  
│ ├── quantum/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── quantum\_chemistry.py  
│ │ └── orbital\_calculations.py  
│ ├── reaction/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── reaction\_pathways.py  
│ │ └── transition\_states.py  
│ └── visualization/  
│ ├── \_\_init\_\_.py  
│ ├── molecular\_visualizer.py  
│ └── reaction\_plotter.py  
├── examples/  
│ ├── molecular\_examples.py  
│ ├── quantum\_examples.py  
│ └── reaction\_examples.py  
├── tests/  
│ ├── test\_molecular.py  
│ ├── test\_quantum.py  
│ └── test\_reaction.py  
├── docs/  
│ ├── chemistry\_guide.md  
│ └── application\_examples.md  
├── setup.py  
├── requirements.txt  
├── README.md  
└── LICENSE

## Installation & Usage

### Prerequisites

* Python 3.8+
* pip package manager
* Git (for development)

### Basic Installation

# Install individual packages  
pip install tsams-core  
pip install tsams-classical  
pip install tsams-cryptography  
pip install tsams-chemistry  
  
# Or install all packages at once  
pip install tsams-core tsams-classical tsams-cryptography tsams-chemistry

# Install individual packages  
pip install tsams-core  
pip install tsams-classical  
pip install tsams-cryptography  
pip install tsams-chemistry  
  
# Or install all packages at once  
pip install tsams-core tsams-classical tsams-cryptography tsams-chemistry

### Development Installation

# Clone repositories  
git clone https://github.com/ctibedoJ/tsams-core.git  
git clone https://github.com/ctibedoJ/tsams-classical.git  
git clone https://github.com/ctibedoJ/tsams-cryptography.git  
git clone https://github.com/ctibedoJ/tsams-chemistry.git  
  
# Install in development mode  
cd tsams-core  
pip install -e .

# Clone repositories  
git clone https://github.com/ctibedoJ/tsams-core.git  
git clone https://github.com/ctibedoJ/tsams-classical.git  
git clone https://github.com/ctibedoJ/tsams-cryptography.git  
git clone https://github.com/ctibedoJ/tsams-chemistry.git  
  
# Install in development mode  
cd tsams-core  
pip install -e .

## Integration Examples

### Cross-Repository Integration

# Combining core and classical components  
from tsams\_core.cyclotomic import CyclotomicField  
from tsams\_classical.prime import PrimeDistribution  
  
# Create a cyclotomic field  
field = CyclotomicField(conductor=7)  
  
# Use prime distribution with cyclotomic field  
prime\_dist = PrimeDistribution()  
result = prime\_dist.analyze\_with\_cyclotomic(field)

# Combining core and classical components  
from tsams\_core.cyclotomic import CyclotomicField  
from tsams\_classical.prime import PrimeDistribution  
  
# Create a cyclotomic field  
field = CyclotomicField(conductor=7)  
  
# Use prime distribution with cyclotomic field  
prime\_dist = PrimeDistribution()  
result = prime\_dist.analyze\_with\_cyclotomic(field)

### Cryptographic Application

# Using ECDLP solver  
from tsams\_cryptography.ecdlp import ECDLPSolver  
from tsams\_cryptography.utils import EllipticCurve  
  
# Define curve parameters  
curve = EllipticCurve(a=2, b=3, p=97)  
  
# Create and run solver  
solver = ECDLPSolver(curve)  
result = solver.solve(point\_p=(3, 6), point\_q=(80, 10))

# Using ECDLP solver  
from tsams\_cryptography.ecdlp import ECDLPSolver  
from tsams\_cryptography.utils import EllipticCurve  
  
# Define curve parameters  
curve = EllipticCurve(a=2, b=3, p=97)  
  
# Create and run solver  
solver = ECDLPSolver(curve)  
result = solver.solve(point\_p=(3, 6), point\_q=(80, 10))

### Chemistry Application

# Molecular modeling with quantum components  
from tsams\_chemistry.molecular import MolecularModel  
from tsams\_chemistry.quantum import QuantumChemistry  
  
# Create molecular model  
model = MolecularModel("C6H6") # Benzene  
  
# Apply quantum chemistry calculations  
qc = QuantumChemistry()  
energy = qc.calculate\_energy(model)

# Molecular modeling with quantum components  
from tsams\_chemistry.molecular import MolecularModel  
from tsams\_chemistry.quantum import QuantumChemistry  
  
# Create molecular model  
model = MolecularModel("C6H6") # Benzene  
  
# Apply quantum chemistry calculations  
qc = QuantumChemistry()  
energy = qc.calculate\_energy(model)

## Contributing

1. Fork the repository
2. Create a feature branch: git checkout -b feature-name
3. Commit changes: git commit -m 'Add feature'
4. Push to branch: git push origin feature-name
5. Submit a pull request

git checkout -b feature-name

git commit -m 'Add feature'

git push origin feature-name

## Documentation

Each repository contains detailed documentation in the docs/ directory:

docs/

* API references
* Mathematical foundations
* Usage examples
* Application guides

## Testing

Run tests for each repository:

cd tsams-core  
python -m pytest  
  
cd tsams-classical  
python -m pytest  
  
cd tsams-cryptography  
python -m pytest  
  
cd tsams-chemistry  
python -m pytest

cd tsams-core  
python -m pytest  
  
cd tsams-classical  
python -m pytest  
  
cd tsams-cryptography  
python -m pytest  
  
cd tsams-chemistry  
python -m pytest

## Roadmap

1. Complete implementation of tsams-biology repository
2. Enhance integration between repositories
3. Develop comprehensive documentation and tutorials
4. Create unified API for cross-disciplinary applications
5. Implement performance optimizations for large-scale computations

This technical guide provides detailed information on the structure, installation, and usage of the TSAMS repositories on GitHub, facilitating immediate adoption and contribution by the community.