C++ code 2.14.24: Sub-problem (2-14.u): Implementation of `solveCRNeumannBVP()`, ➙ GitLab

```cpp
template <typename GAMMA_COEFF, typename F_FUNCTOR>
Eigen::VectorXd solveCRNeumannBVP(std::shared_ptr<CRFeSpace> fe_space,
                                  GAMMA_COEFF &&gamma, F_FUNCTOR &&f) {
  Eigen::VectorXd sol;
// TODO: task 2-14.u)
  // Obtain local to global index mapping for shape functions
  const lf::assemble::DofHandler &dof_handler{fe_space->LocGlobMap()};
  const size_type num_dofs = dof_handler.NumDofs();
  // Prepare coefficient and source functions as MeshFunction
  lf::mesh::utils::MeshFunctionGlobal mf_one{
      [](Eigen::Vector2d x) -> double { return 1.; }};
  lf::mesh::utils::MeshFunctionGlobal mf_gamma{gamma};
  lf::mesh::utils::MeshFunctionGlobal mf_f{f};
  // Sparse Galerkin matrix in triplet format
  lf::assemble::COOMatrix<double> A(num_dofs, num_dofs);
  // Initialize ELEMENT_MATRIX_PROVIDER object
  lf::uscalfe::ReactionDiffusionElementMatrixProvider<double, decltype(mf_one),
                                                      decltype(mf_gamma)>
      element_matrix_builder(fe_space, mf_one, mf_gamma);
  // Fill Galerkin matrix (create array of triplets)
  lf::assemble::AssembleMatrixLocally(0, dof_handler, dof_handler,
                                      element_matrix_builder, A);
  // Right-hand-side vector; do not forget to set to zero!
  Eigen::Matrix<double, Eigen::Dynamic, 1> phi(num_dofs);
  phi.setZero();
  // Initialize ELEMENT_VECTOR_PROVIDER object
  lf::uscalfe::ScalarLoadElementVectorProvider<double, decltype(mf_f)>
      load_vector_builder(fe_space, mf_f);
  // Fill right-hand-side vector (cell oriented assembly)
  lf::assemble::AssembleVectorLocally(0, dof_handler, load_vector_builder, phi);
  // Set up Galerkin matrix in CRS format
  Eigen::SparseMatrix<double> A_crs = A.makeSparse();
  // ... and solve the linear system of equations by Gaussian elimination
  Eigen::SparseLU<Eigen::SparseMatrix<double>> solver;
  solver.compute(A_crs);
  sol = solver.solve(phi);
  return sol;
}
```