Employee Payroll Processing and Management System

Team Task Assignment (Group of 4)

Below are individual role assignments and clear, actionable tasks derived from the project documentation you provided. Each member has responsibilities, subtasks, recommended file/module names, and deliverables. Use these as the working plan for development, testing, documentation, and presentation.

Princeton (You)

Roles: Project Lead, Core Programmer / Integration

- Coordinate the team: run weekly check-ins and track progress (record short status notes).
- Finalize and maintain the master project schedule and repository structure (git or zipped folders).
- Develop core modules: main program flow, batch payroll runner, file locking/backup manager.
- Integrate modules from other devs and ensure consistent file formats.
- Prepare the demo script and perform final integration testing before demo.
- Deliverables: master repo (source code), batch_processor.py or batch_processor.c, integration README.

Jessica

Roles: System Analyst, Documentation Officer

- Refine and finalize the data model and file schema (employee file, payroll file, backup file).
- Produce detailed data dictionaries (field names, types, sizes) and flowcharts for each module.
- Complete and polish all PDF documentation we already generated (SRS, System Design, User Manual) with module-level details.
- Write the user manual's step-by-step examples and the 'how to run' section.
- Prepare final technical report and coordinate with Princeton for any required changes.
- Deliverables: data_dictionary.pdf, updated_SRS.pdf, final_user_manual.pdf.

Philip

Roles: Programmer / Developer (Payroll Logic & File I/O)

- Implement Employee Management CRUD (create, read, update, delete) operations.
- Implement payroll computation functions: gross pay, allowances, taxes, statutory deductions, net pay.
- Build CSV/binary file readers and writers and an index lookup if required.
- Write unit tests for payroll calculations (use sample records and edge cases).
- Deliverables: employee_crud.py or employee_crud.c, payroll_logic.py or payroll_logic.c, unit_tests.md or tests/ folder.

Richard

Roles: QA Officer, Presentation Lead

- Design and run test cases covering CRUD operations, payroll computations, batch runs, and backup/restore.
- Document test results and log any defects (include reproduction steps and severity).
- Verify file locking behavior and simulate concurrent access scenarios where practical.
- Build the presentation slides and coordinate demo flow with Princeton.
- Deliverables: test_report.pdf, bug_log.csv, presentation_slides.pdf.

Shared Responsibilities & Checkpoints

- Use a shared repository (GitHub, GitLab, or a shared cloud folder). Keep a simple README with build/run instructions.
- Naming convention: use clear file names (example: payroll_logic.py, employee_crud.py, batch_processor.py).
- Weekly checkpoint deliverable: short status update + one working artifact (code file, pdf, or test log).
- Backup your work frequently; place final versions in a dedicated /final folder before submission.
- All code should include comments and a small README for how to run that module.

Suggested Milestones (for internal tracking)

- Milestone 1 — Setup repo, finalize data dictionary, and create skeleton code files.
- Milestone 2 — Complete CRUD and payroll logic modules; basic unit tests complete.
- Milestone 3 — Implement batch processing, backup & record-locking; integration testing.
- Milestone 4 — Final testing, prepare presentation, and package deliverables.

Acceptance Criteria (What counts as 'done')

- All core features (CRUD, payroll calc, batch run, backup/restore) implemented and tested.
- Documentation (SRS, Design, Test Report, User Manual, Presentation) finalized and in PDFs.
- Demo runs smoothly showing add employee → batch payroll → report generation → backup restoration.
- Code is commented and organized; team can explain their modules during presentation.

If you'd like, I can now (a) generate individual task checklists in separate PDFs per member, or (b) create starter code templates for each assigned module. Tell me which you prefer and I’ll do it next.