```
class App {
  constructor(employeeTable, pageHeader, employeeForm){
    this.employeeTable = employeeTable;
    this.pageHeader = pageHeader;
    this.employeeForm = employeeForm;
    this.employees = [];
    this.nextId = 1000; // Starting ID for new employees (using higher number to avoid conflicts)

    this.computeEmployeeCount = this.computeEmployeeCount.bind(this);
    this.editEmployee = this.editEmployee.bind(this);
    this.getEmployees = this.getEmployees.bind(this);
    this.createEmployee = this.createEmployee.bind(this);
    this.deleteEmployee = this.deleteEmployee.bind(this);
    this.updateEmployee = this.updateEmployee.bind(this);
  }

  start() {
    this.getEmployees();
    this.employeeForm.onSubmit(this.createEmployee);
    this.employeeForm.onUpdate(this.updateEmployee);
    this.employeeTable.onDeleteClick(this.deleteEmployee);
    this.employeeTable.onEditClick(this.editEmployee);
  }
  computeEmployeeCount(employeesObj) {
    return employeesObj ? employeesObj.length : 0;
  }
  editEmployee(name, lastName, position, department_name, salary, id){
    this.employeeForm.editEmployeeForm(name, lastName, position, department_name, salary,
id);
  }
  addAllEmployees(employees){
    this.employees = employees;
  }
  addAEmployee(employee){
    this.employees.push(employee);
  }
  deleteAEmployee(id){
    if(this.employees){
      for (let i = 0; i < this.employees.length; i++) {
        if(this.employees[i].id == id){
          this.employees.splice(i,1);
        }
      }
    }
```

```
}
updateAEmployee(name, lastName, position, department_name, salary, id){
  if (this.employees) {
    for (let i = 0; i < this.employees.length; i++) {
      if (this.employees[i].id == id) {
        this.employees[i].name = name;
        this.employees[i].lastName = lastName;
        this.employees[i].position = position;
        this.employees[i].department_name = department_name;
        this.employees[i].salary = salary;
      }
    }
  }
}

getEmployees() {
  // Load employees from local JSON file
  fetch('./db.json')
    .then(response => response.json())
    .then(employees => {
      this.addAllEmployees(employees);
      this.employeeTable.updateEmployees(employees);
      this.pageHeader.updateEmployeeCount(this.computeEmployeeCount(employees));
    })
    .catch(error => {
      console.error('Error loading employees:', error);
      // Fallback to empty array if file can't be loaded
      this.addAllEmployees([]);
      this.employeeTable.updateEmployees([]);
      this.pageHeader.updateEmployeeCount(0);
    });
}

createEmployee(name, lastName, position, department_name, salary) {
  // Create new employee object with local data
  const newEmployee = {
    id: this.nextId,
    name: name,
    lastName: lastName,
    position: position,
    department_name: department_name,
    salary: parseInt(salary),
    parentId: "",
    email: "",
      phone_number: "",
      hire_date: new Date().toISOString(),
      job_id: "",
      commission_pct: "",
      department_id: 0,
      job_min_salary: 0,
      location_state: "",
      job_max_salary: 0,
      department_location_id: 0,
      department_location_street_address: "",
      department_location_postal_code: 0,
      department_location_country_id: "",
      department_location_country_name: "",
      department_location_country_region_id: 0,
      department_location_country_region_name: "",
      image: "https://bumbeishvili.github.io/avatars/avatars/portrait1.png"
    };

    this.nextId++; // Increment for next employee
    this.addAEmployee(newEmployee);
    this.employeeTable.updateEmployees(this.employees);
    this.pageHeader.updateEmployeeCount(this.computeEmployeeCount(this.employees));
  }

  deleteEmployee(id){
    this.deleteAEmployee(id);
    this.employeeTable.updateEmployees(this.employees);
    this.pageHeader.updateEmployeeCount(this.computeEmployeeCount(this.employees));
  }

  updateEmployee(name, lastName, position, department_name, salary, id) {
    this.updateAEmployee(name, lastName, position, department_name, salary, id);
    this.employeeTable.updateEmployees(this.employees);
    this.pageHeader.updateEmployeeCount(this.computeEmployeeCount(this.employees));
  }
}
```