**Backend Development Repository Structure:**

* The Back end is mainly a Flask Rest API
* The api\_module will handle all the call from the client side
* The file structure is represented as follows :

│   ├── api\_module

│   │   ├── \_\_init\_\_.py

│   │   ├── chat\_controller.py

│   │   ├── company\_controller.py

│   │   ├── employee\_controller.py

│   │   ├── helpers.py

│   │   ├── issue\_controller.py

│   │   ├── models.py

│   │   ├── project\_controller.py

│   │   ├── role\_controller.py

│   │   ├── sprint\_controller.py

│   │   ├── task\_controller.py

│   │   ├── team\_controller.py

│   │   └── user\_controllers.py

│   └── templates

│   ├── 404.html

│   └── docstring.html

├── config.py

├── local

│   └── db

│   └── app.db

├── readme.md

├── run.py

├── static

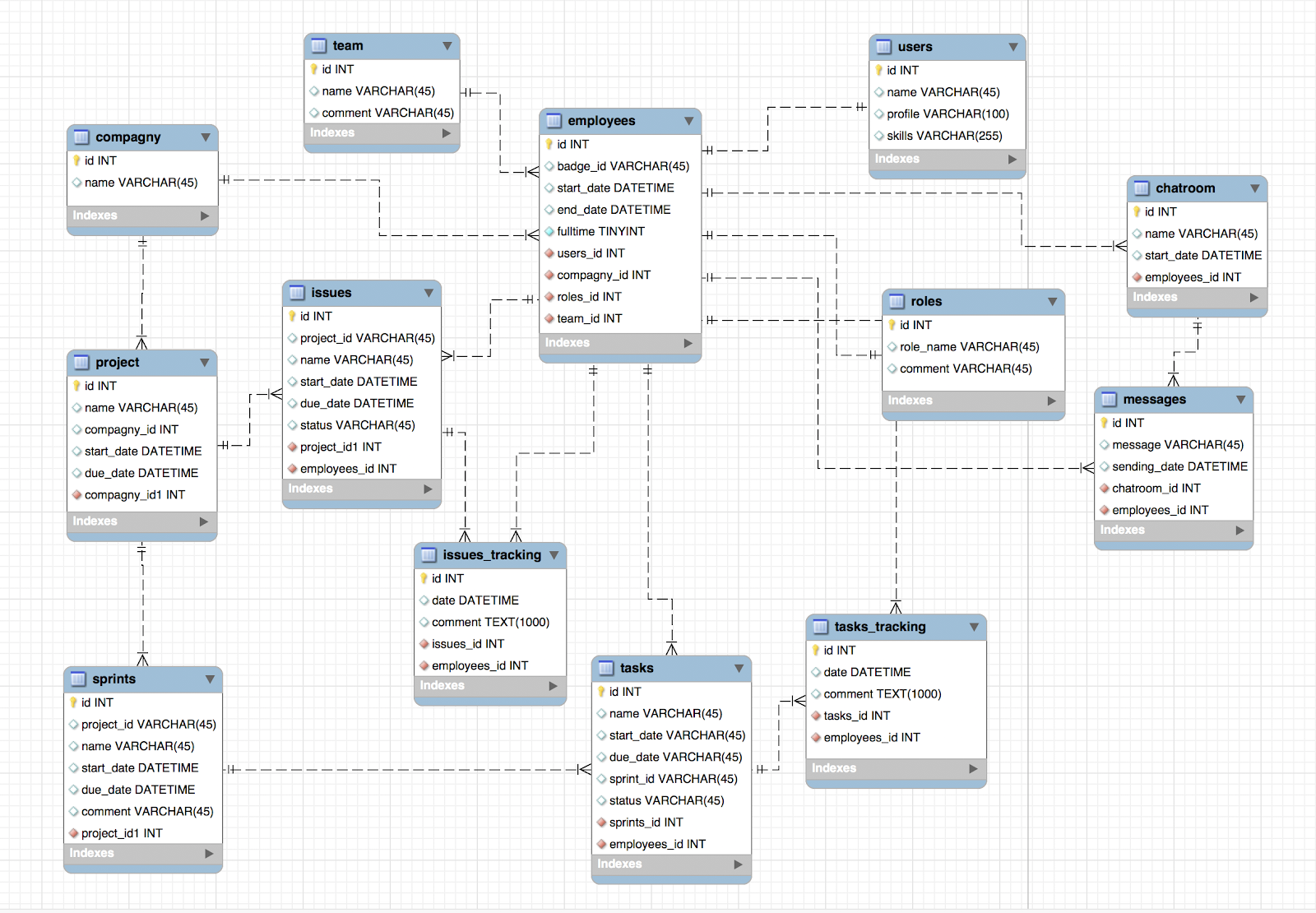
└── tests

├── payload.txt

└── pmp\_backend.postman\_collection.json

* Run a development server using the run.py
  + GET : <http://0.0.0.0:5005/api/>
* app.db : SQLite Database created by models.py

**Database Model: E-R Diagram:**



**Backend controller job description:**

* Company\_controller.py: Controller Job to create a new company, fetch specific company and fetch list of companies.
* Employee\_Controller.py: Create employee controller job to create new employee under a company, fetch all employees under a company, and fetch all employees under a team.
* Issue\_Controller.py: Create issue controller job to:

- create new issues under project, with employee name, start date and due date

- Update the issue for due date and status

- Get all the issues assigned to a user

- Get all the issues under a project

- Get all the issues for a project filter by status

- Create comments under issue tracking

- Delete comments from issue tracking

* Project\_Controller.py: Create project controller job to create new projects, fetch a project, fetch all projects within a company, and throw warning if searched project is not found.
* Role\_Controller.py: Create role.py job to create role for a user id, get all the roles for a user id, get specific role for a user id
* Sprint\_Controller.py: Create sprint\_controller.py job to create a sprint, get all the sprints under a project, get all the sprints under a user, get specific sprint under project.
* Task\_Controller.py: Create tasks controller job to:

- create new tasks under sprint, with employee name, start date and due date

- Update the tasks for due date and status

- Get all the tasks for a user

- Get all the tasks for under a sprint

- Get all the tasks for a sprint filter by status

- Create comments under task tracking

- Delete comments from task tracking

* Team\_Controller.py: Create team controller job to create new teams under a company, fetch all teams for a user and fetch one team under a user id
* User\_Controllers.py: Create user\_controllers.py python file to create new users ids, get and delete existing user ids.