By :- Solasa Manikanta Kumar

FoodMeOnce



About the Site

* **4 sprints**
  + *Phase 1*: Basic Static Website
  + *Phase 2*

■ Dynamic React Web Application

■ RESTful API

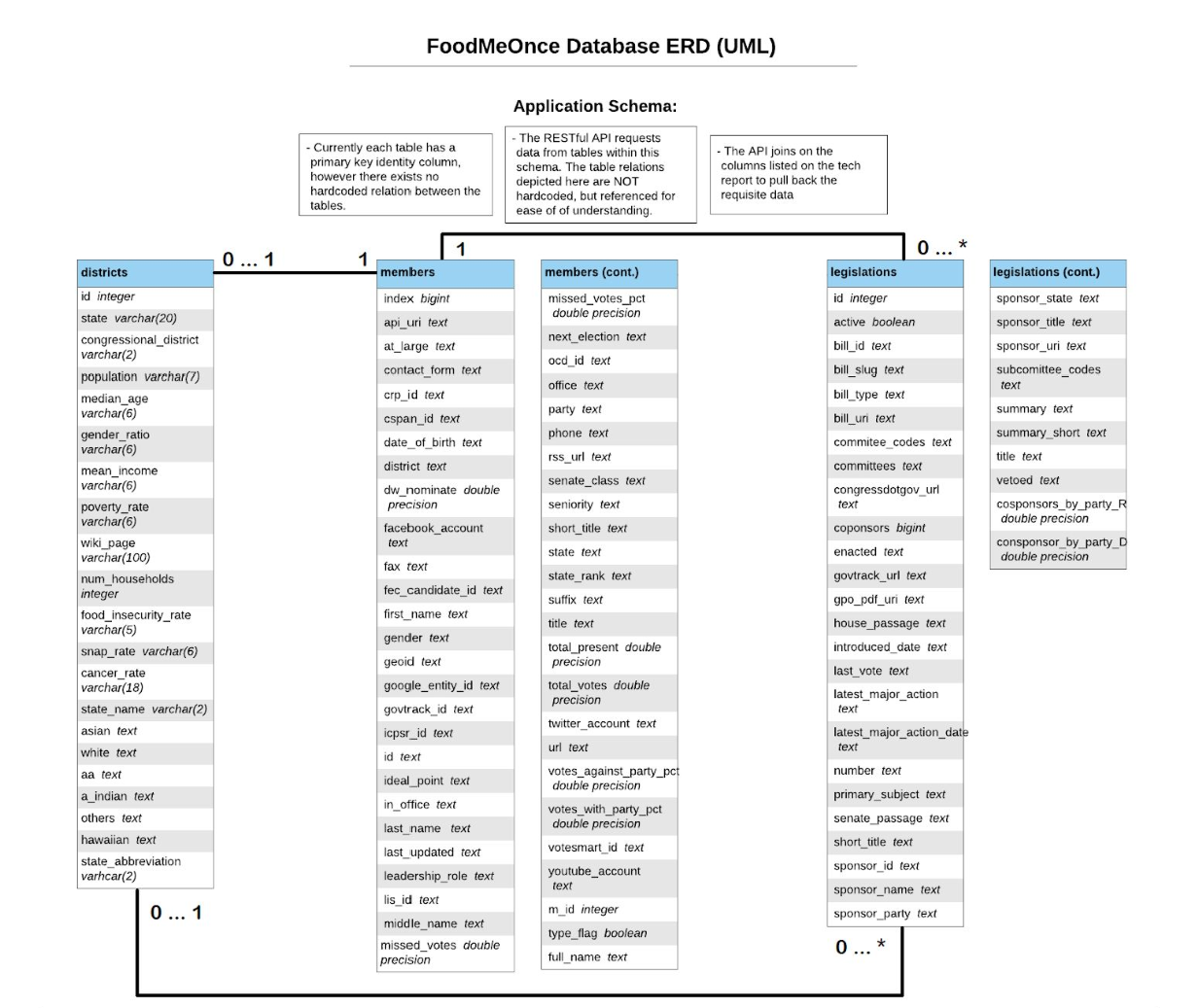
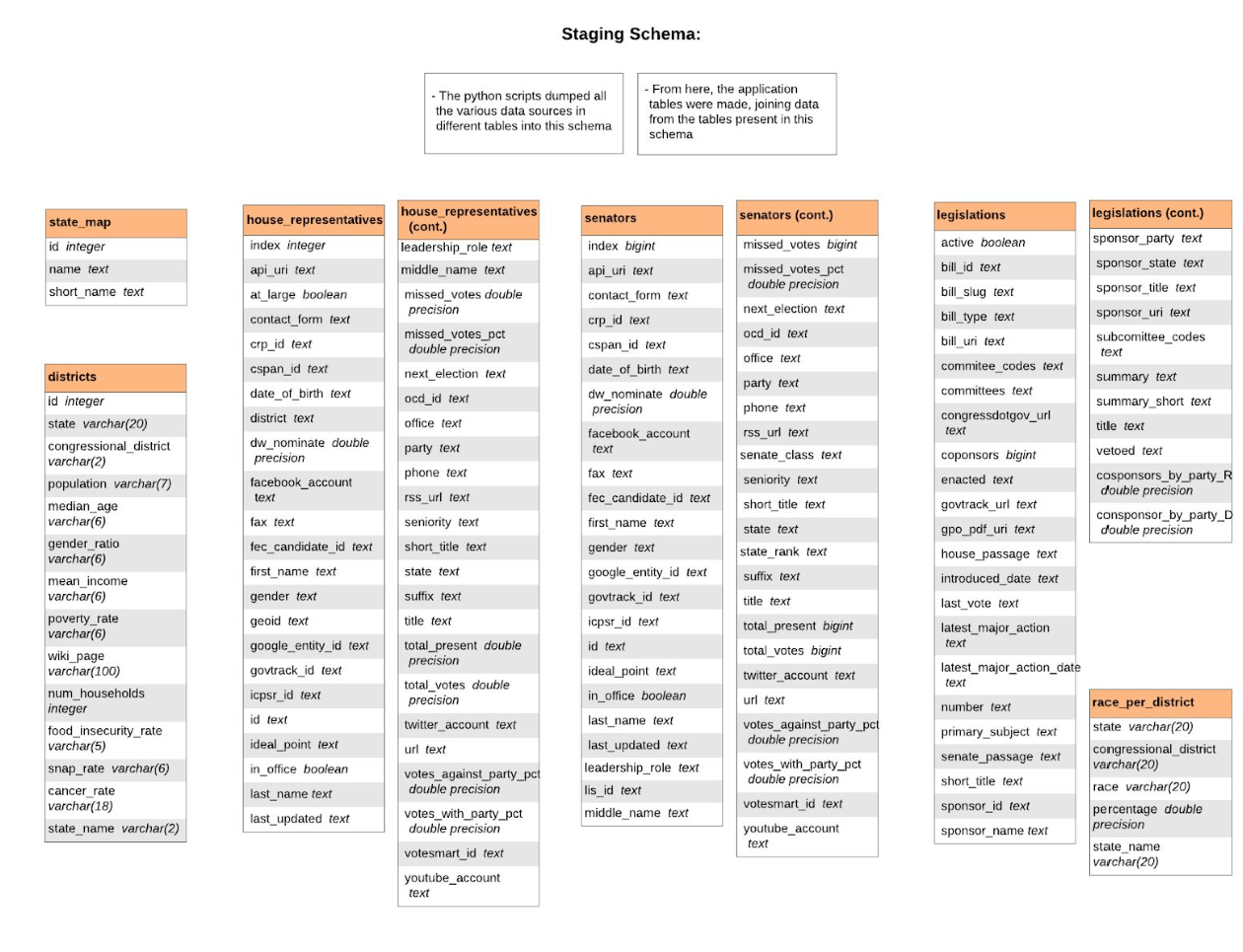
■ PostGres DB

* + *Phase 3*: Sorting, Searching, Filtering
  + *Phase 4*: Visualizations
* **Models**
  + *Districts*
  + *Representatives*
  + *Legislation*
* **Links**
  + FoodMeOnce GitLab Repository
  + FoodMeOnce API Documentation

What is FoodMeOnce?

* Web Platform to allow users to gather information on food security throughout US Congressional Districts
* By combining Disparate Data sources we provide a well rounded perspective on food security in relation to political representation and legislation
* Generates Easy to understand Visualizations to allow a quick grasp of Congressional food support using various dimensions such as population, race, and representation.

Database



Tool stack

**Front-end frameworkBack-End ToolsBackend**

* React Javascript ❖ PostgreSQL❖ Amazon S3
* Bootstrap and CSS ❖ POSTMAN**Domain**
* Selenium❖ SQLAlchemy❖ NameCheap
* Mocha❖ Flask❖ Route53 ❖ D3❖ Python**Others**

**IDEs**❖ Gitlab

* Pycharm IDE❖ LucidChart
* VSCode❖ Docker
* Gitlab CI/CD

**Demonstration**



What did we do well?

* Made website dynamic in phase 1
* Searching, Sorting, Filtering ❖ Getting each phase reviewed by the TAs
* Well paced throughout each phase of the project.
* Gitlab Issue Board
* Communication with our Customer team

What did we learn?

* E2E website building
* Deploying and hosting in AWS
* Designing an API
* Domains and subdomains
* Database design
* React and Flask frameworks
* Postman API documentation
* Importance of code readability

What can we do better?

* UI/UX
  + Relative spacing of elements on screen
  + Did not consider mobile usage
  + Prettier visuals
  + Splash page
* Code refactoring/maintainability
* Could have used flask-restless
* District model page - dynamic map load time

What puzzled us?

* How to get started
* Asynchronous API requests
* Searching Data
* D3
* Gitlab CI setup

Developer Team - PutItInPark



What did they do well?

* UI/UX is great
  + On mouse hovering over cards
  + Overall look and feel
  + Buttons
  + Visualizations located on model pages
  + Pagination has 5 pages
* Communicating on user stories was very thorough
* Made site mobile friendly

What did we learn from their website?

Technical gains:

* Hovering effect over cards
* Using entire space for the instance page
* Use of icons(search icon, loading icon)

Website specific:

* Central USA has the most frequency of parks + recreational activities

What can they do better?

* Meeting phase requirements
  + API subdomain
* Model page search results sometimes do not contain search term
  + National Parks search

What puzzles us?

Technical puzzles:

* How the model specific search works?
  + Possibly not correctly matching (ex) acadia in National Parks and tenn in States
  + No highlighting search terms on model specific search
* Website down at one point

Website specific:

* Why does Central USA have such low visitor count despite having the most parks + recreational activities?

Questions?

Thank you!