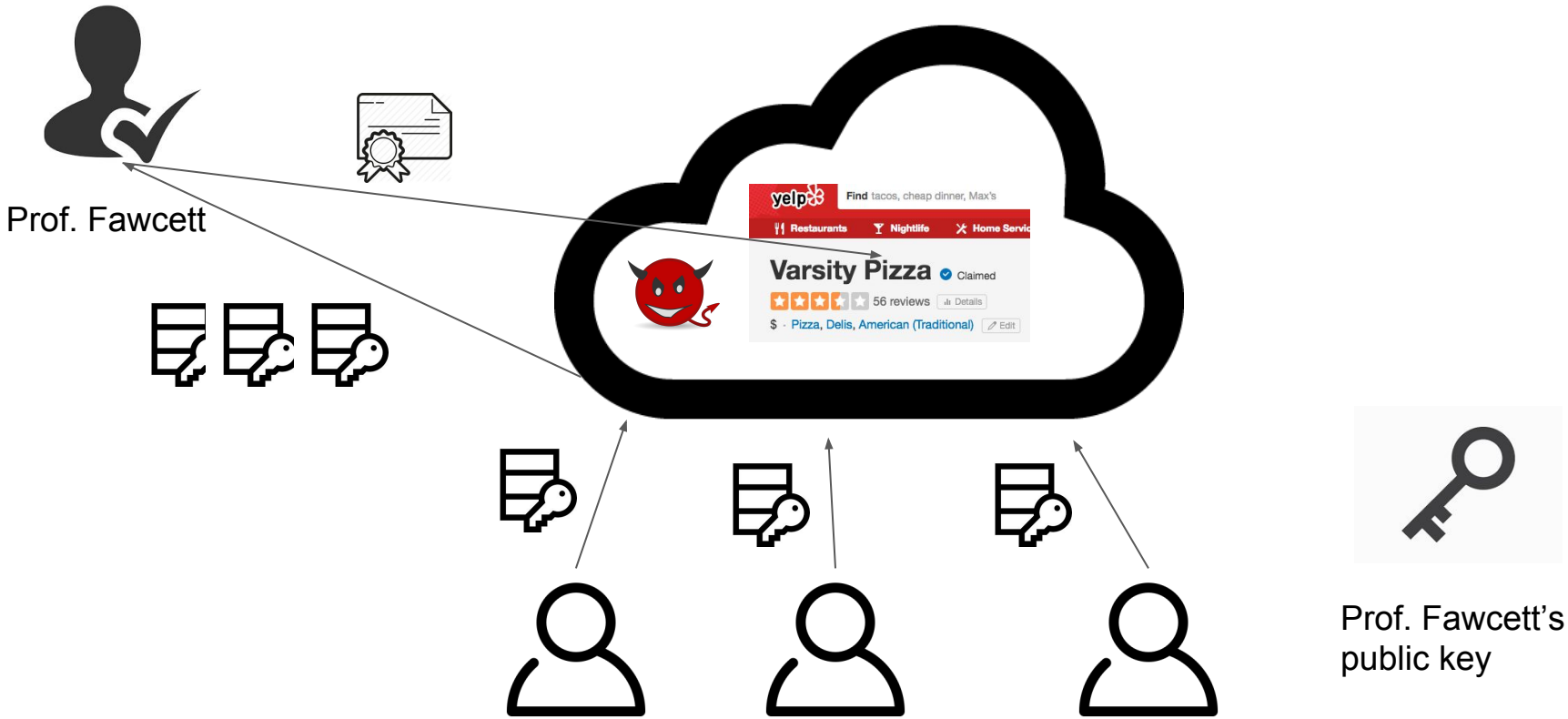
Secret Voting

Using MEAN Stack

Ju Chen, D.O. final project (Spring 2017) - Probing examples

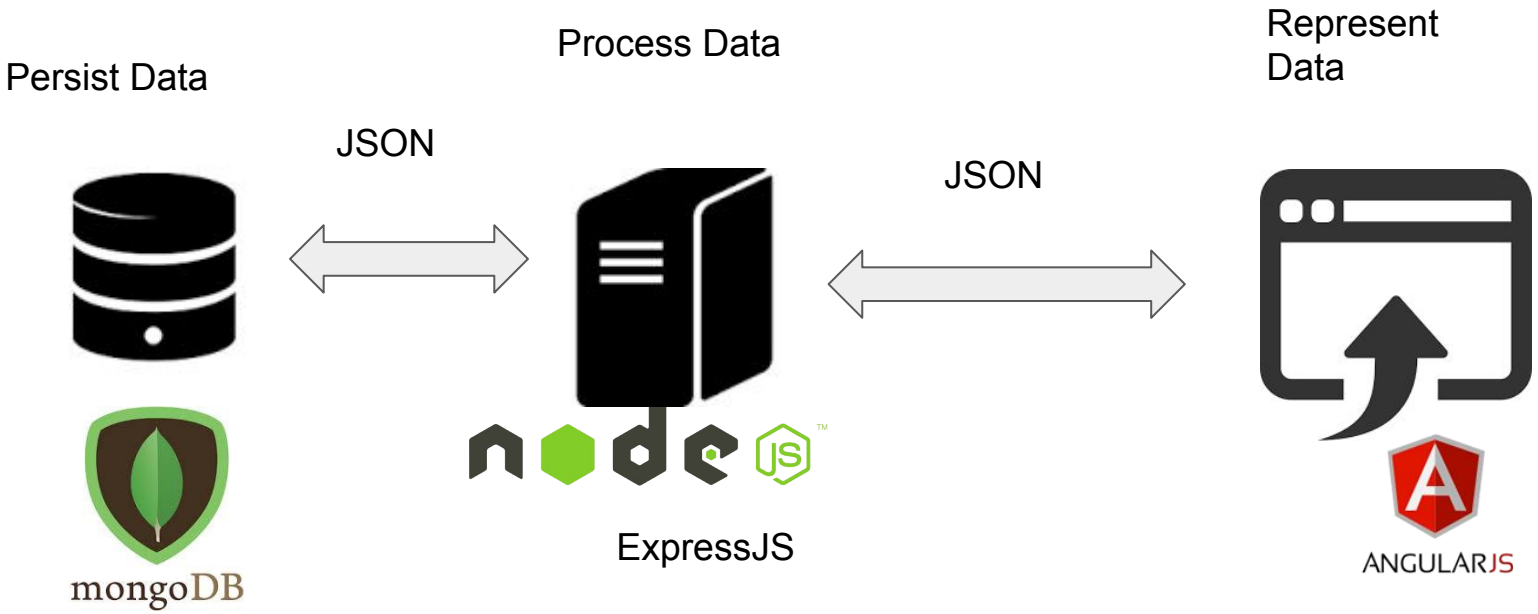
Idea - Encrypt, Offload and Sign



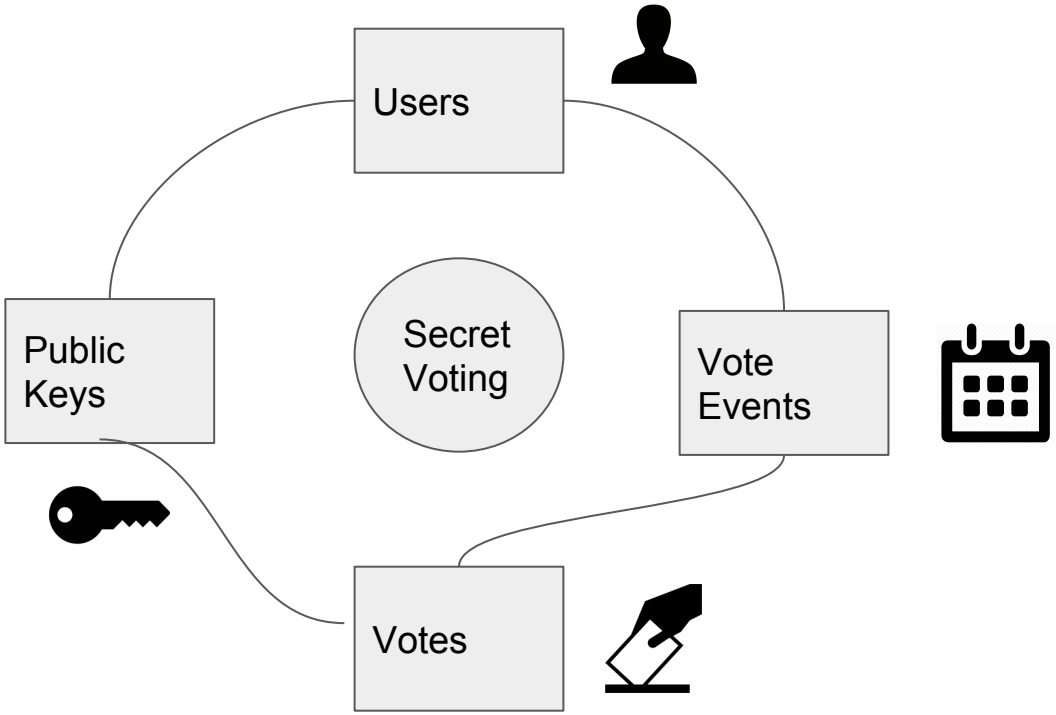
Project Deliverables

- A small voting website doing voting
  - User Login
  - Roles (Admin, Voter, Scrutineer)
  - Trusted-peer publish its Public Key
  - Anyone can initialize a vote event
  - Vote and Submit (Using openssl lib to do the encryption)
  - Trusted-peer receives request from server to count the votes
  - Trusted-peer submits and signs the results. The result is displayed when the event is completed

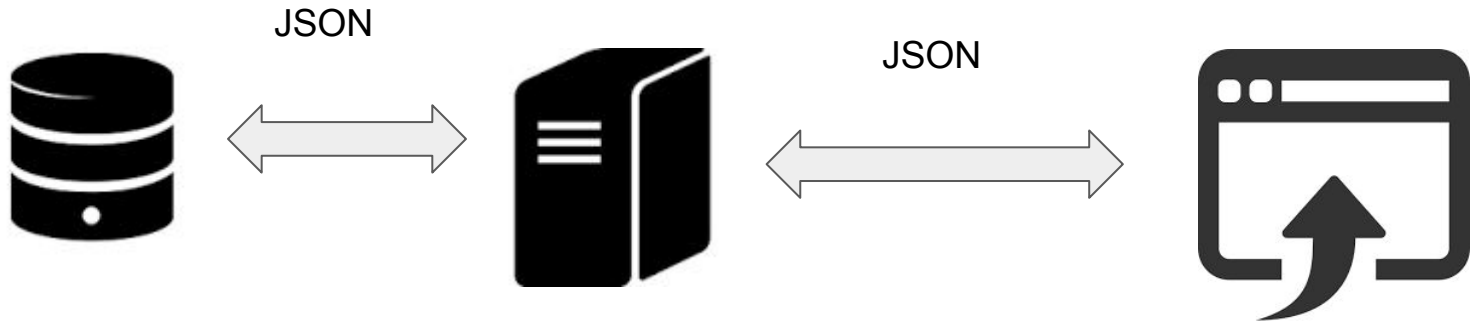
Mean stack (One Language, One Data Type)

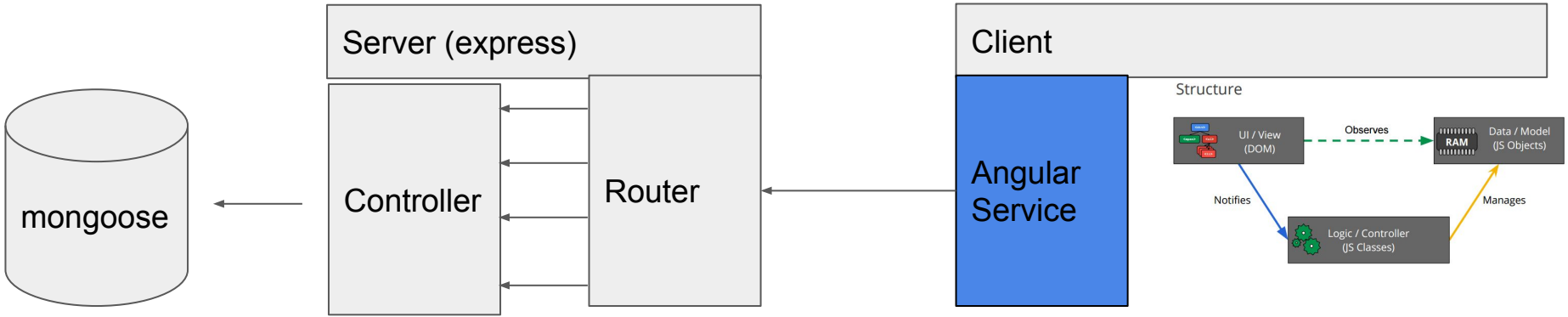


Project Horizontal Structure

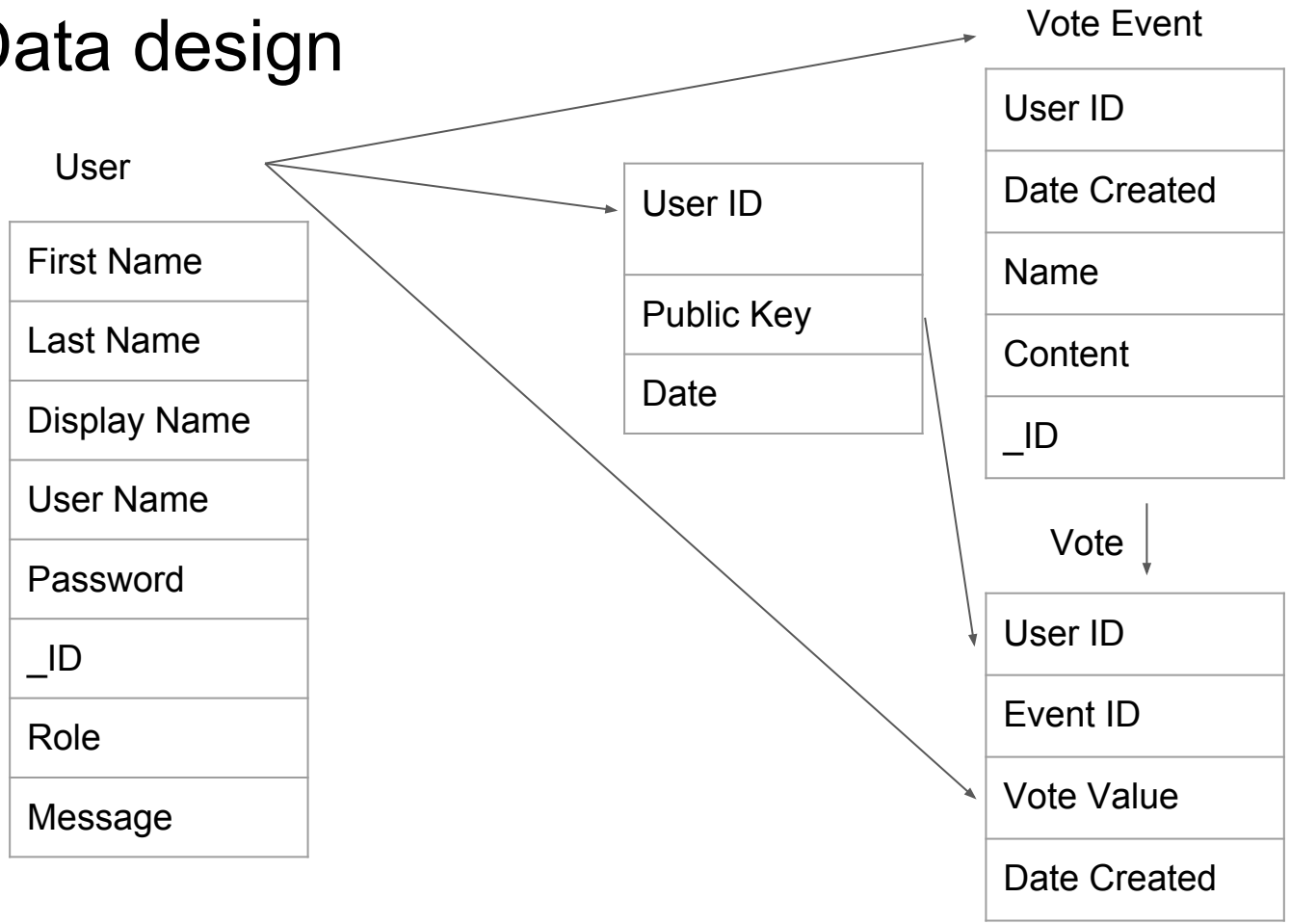


Project Vertical Structure (for one module)





Data design


User
First Name
Last Name
Display Name
User Name
Password
_ID
Role
Message
User ID
Public Key
Date
Vote Event
User ID
Date Created
Name
Content
_ID
Vote
User ID
Event ID
Vote Value
Date Created

Code examples and Running Examples

Server Side Controllers

```
13 /**
14  * Create a Vote event
15  */
16 exports.create = function(req, res) {
17   var voteEvent = new VoteEvent(req.body);
18   voteEvent.user = req.user;
19   voteEvent.save(function(err) {
20     if (err) {
21       return res.status(400).send({
22         message: errorHandler.getErrorMessage(err)
23       });
24     } else {
25       res.jsonp(voteEvent);
26     }
27   });
28 };
```

Server Side Routers

```
1  'use strict';
2
3  /**
4   * Module dependencies
5   */
6  var voteEventsPolicy = require('../policies/vote-events.server.policy'),
7    voteEvents = require('../controllers/vote-events.server.controller');
8
9  module.exports = function(app) {
10   // Vote events Routes
11   app.route('/api/vote-events').all(voteEventsPolicy.isAllowed)
12     .get(voteEvents.list)
13     .post(voteEvents.create);
14
15   app.route('/api/vote-events/:voteEventId').all(voteEventsPolicy.isAllowed)
16     .get(voteEvents.read)
17     .put(voteEvents.update)
18     .delete(voteEvents.delete);
19
20   // Finish by binding the Vote event middleware
21   app.param('voteEventId', voteEvents.voteEventByID);
22 };
23
```

Code examples and Running Examples

Client side service (call service side APIs)

```
 4 angular
 5 .module('vote-events')
 6 .factory('VoteEventsService', ['$resource',
 7     function ($resource) {
 8     return $resource('api/vote-events/:voteEventId', {
 9         voteEventId: '@_id'
10     }, {
11         update: {
12         method: 'PUT'
13         }
14     });
15 }
16 ]);
```

Client side service can be cross-referenced

```
3 // Articles controller
4 angular.module('vote-events').controller('VoteEventsController', ['$scope',
5 '$stateParams', '$location', 'Authentication', 'VoteEventsService', 'VotesService',
6     function ($scope, $stateParams, $location, Authentication, VoteEventsService, VotesService) {
```

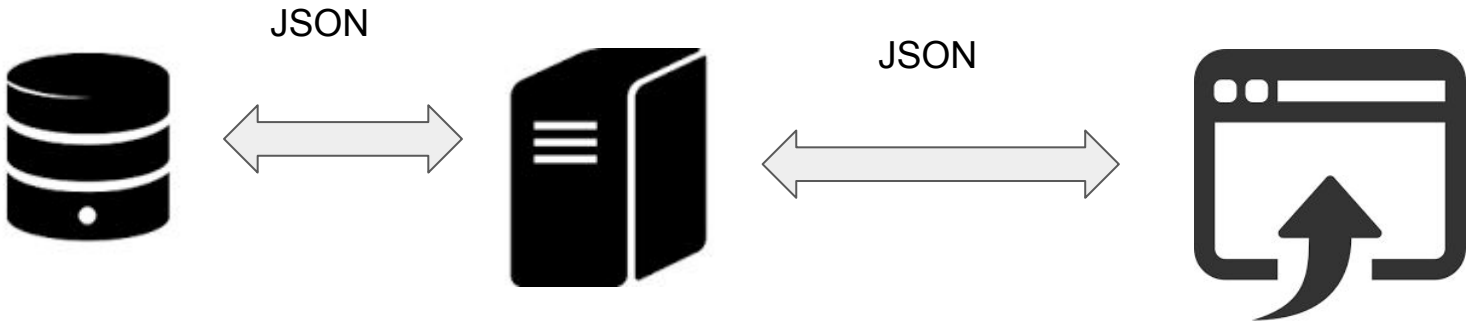
```
2 'use strict';
3
4   angular
5     .module('votes')
6     .factory('VotesService', ['$resource',
7
8   function ($resource) {
9     return $resource('api/votes/:voteId', {
10      voteId: '@_id'
11    }, {
12      update: {
13        method: 'PUT'
14      }
15    });
16  }
17 ]);
```

Client side controller retrieves the data

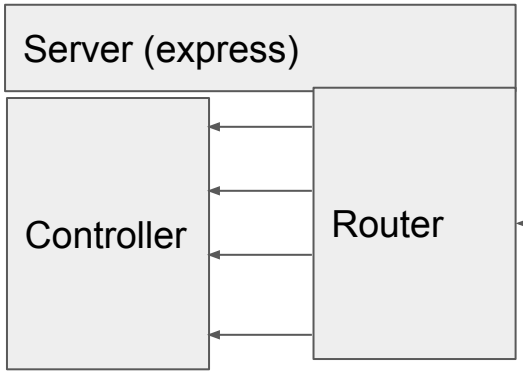
```
3 // Articles controller
4 angular.module('vote-events').controller('VoteEventsListController', ['$scope', '$stateParams',
5   function ($scope, $stateParams, $location, Authentication, VoteEventsService) {
6     $scope.authentication = Authentication;
7
8     // Find a list of Articles
9     $scope.find = function () {
10      $scope.voteEvents = VoteEventsService.query();
11    };
12  }
13 ]);
```

Angular Data Binding helps to present the data

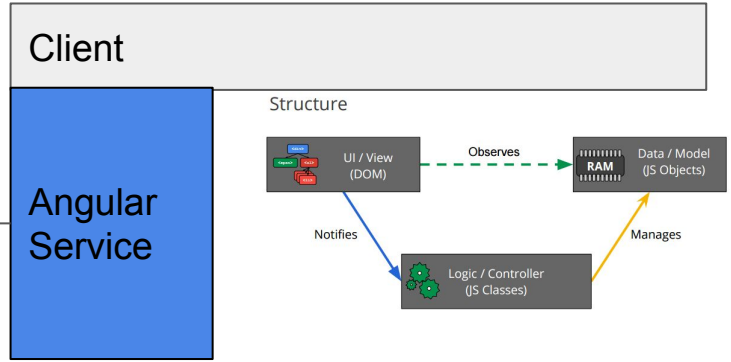
```
<section data-ng-controller="VoteEventsListController" data-ng-init="find()">
  <div class="page-header">
    <h1>Vote events</h1>
  </div>
  <div class="list-group">
    <div data-ng-repeat="voteEvent in voteEvents"
       class="list-group-item">
      <small class="list-group-item-text">
        Posted on
        <span data-ng-bind="voteEvent.created | date:'mediumDate'"></span>
        by
        <span data-ng-if="voteEvent.user" data-ng-bind="voteEvent.user.displayName"></span>
        <span data-ng-if="!voteEvent.user">Deleted User</span>
      </small>
      <h4 class="list-group-item-heading" data-ng-bind="voteEvent.name"></h4>
<div class="btn label-success" data-ng-if="voteEvent.state=='open'">Voting is on-going!</div>
<div class="btn label-danger" data-ng-if="voteEvent.state=='close'">The event is over</div>
<div class="btn btn-primary" data-ng-show="authentication.user._id==voteEvent.user._id" data-ui-sref="vote-events.view({ voteEventId:
Edit</div>
<div class="btn btn-primary" data-ui-sref="vote-events.vote({voteEventId:voteEvent._id})">Vote For this</div>
    </div>
  </div>
  <div class="alert alert-warning text-center" data-ng-if="voteEvents.$resolved && !voteEvents.length">
    No Vote events yet, why don't you <a data-ui-sref="vote-events.create">create one</a>?
  </div>
</section>
```



Vote Events



Vote Events



Running Example

Debug Techniques

1, In front end, use alert()

2, In back end, use console.log()

3, To view data in database, use mongo

4, To test the RESTFul API, use browser or other advanced tools

Cryptography Part

1, OpenSSL demo

2, JavaScript OpenSSL demo

Conclusion

- MEAN stack development
  - JSON data centric
  - Mongoose API
  - Express (Router and Controller)
  - AngularJS (Data-binding, Service, Presentation Skill)
- Website design strategy
  - Design Data Model first
  - Components Decomposition
  - Communication
  - Design Views
  - Connect data and views
- Cryptography

Thank you!