Interview Challenge

The challenge consists in create 2 applications, one in Angular and another in Java with SpringBoot.

Both applications will have the same user stories, and each layer should create a design and a solution for it.

Middleware

The middleware developer should create an API design to cover all the detected user cases, also should create a quick implementation of the SpringBoot application.

We recommend use a H2 database for persistence and simulate the interaction with a relational database.

All the Spring Framework is suitable to be used.

Front-End

The angular developer should create a web application to cover all the detected user cases.

The developer should create a diagram with the components, explaining the relationships among them.

For the API we provide an openAPI (swagger) file as API contract. THIS DESIGN IS ONLY FOR FRONT-END Developer, NEVER have to be shown to middleware developer because create it is part of the middleware challenge.

STORIES

USER

Feature: Session management

```
Scenario: New session
 Given I want create a place to make a poker planning session
When I type a title
 And select the deck type
 And summit the form
 Then I enter in a new poker planning session
 And I have a link to invite to other people to join

Scenario: Enter session
```

```
 Given I receive an invitation link to join in a poker planning ses
sion
 And I navigate to it
 When I type my name/nickname
 And summit the form
 Then I enter in the poker planning session
 And I see the title of the session
 And I able to see the user stories list to vote
 And I able to see the members list joined in the session

Scenario: Destroy poker planing session
 Given I'm in as poker planning session
 And I want to destroy the session
 When I push "Destroy Session" button
 And double check my intend
 Then All data are destroyed
 And I'm redirected to a confirmation page
```

Feature: Votes management

```
Scenario: Start voting a user story
 Given There are PENDING or VOTED user stories
 When I push the button "Start" of the user story
 Then the user story move to VOTING status
 And Card/Vote options are enable to select for all the connected m
embers

Scenario: Vote a user story
 Given There is a user story in VOTING status
 When I select a card/vote option for it
 Then the user story "emitted votes" increase in 1
And I'm marked as vote emmited
 And the other members are not able to see my vote

Scenario: Listen to the votation status
 Given There is a user story in VOTING status
 When someone emits a vote
 Then all the members see who has emited the vote
 And number of emited votes in the user story
 But Nobody is able to see the value of the vote from another membe
r.

Scenario: Finish voting a user story
 Given There is a user story in VOTING status
When I push the button "Stop" of the user story
Then the user story move to VOTED status
 And no more votes are accepted for the user story  And
the votes of all members are revealed  And summary of
voted values are shown in the user story
```

Feature: User stories management

```
Scenario: Add a user story
 Given I'm in as poker planning session
 And I want to add a user story
 When I fill user story Id and description  Then The
user story is added in the user story list

Scenario: Delete a user story
 Given I'm in as poker planning session
 And I want to delete a user story
 And the user story is PENDING
 When I push the option to delete the user story  Then the
user story disappears from the user story list
```

OpenAPI – Swagger Contract

```
openapi: 3.0.0
info:
  title: Poker Planning API
  description: REST API to support a poker planning
  session version: 0.0.1
paths:
  /sessions:
    get:
      summary: List of existing poker planning sessions
      tags:
        - session management
      responses:
        '200': # status code
          description: A JSON array of session
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: "#/components/schemas/PokerPlanningSession"
    post:
      summary: Creation of a new session
      tags:
        - session management
      requestBody:
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/PokerPlanningSession"
      responses:
        '201': # status code
          description: Created JSON object
          content:
```

```yaml
          application/json:
            schema:
              $ref: "#/components/schemas/PokerPlanningSession"
/sessions/{idSession}:
  parameters:
    - $ref: "#/components/parameters/IdSessionParam"
  get:
    summary: Session information
    tags:
      - session management
    responses:
      '200':
        description: The Resource
        content:
          application/json:
            schema:
              $ref:
      "#/components/schemas/PokerPlanningSession" '404':
          description: Not found
  delete:
    summary: Destroy session
    tags:
      - session management
    responses:
      '200':
        description: Session destroyed info
        content:
          application/json:
            schema:
              $ref:
      "#/components/schemas/PokerPlanningSession" '404':
          description: Not found
/sessions/{idSession}/members:
  parameters:
    - $ref: "#/components/parameters/IdSessionParam"
  get:
    summary: List of Members in the session
    tags:
      - session management
    responses:
      '200':
        description: List of Members join in
        content:
          application/json:
            schema:
              type: array
              items:
                $ref: "#/components/schemas/Member"
      '204':
          description: No content
      '404':
          description: Not found
  post:
    summary: Join in the session
    tags:
      - session management
    requestBody:
      content:
        application/json:
          schema:
            $ref: "#/components/schemas/Member"
    responses:
      '201': # status code
        description: Created JSON object
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/Member"
      '404':
          description: Not found
/sessions/{idSession}/members/{idMember}
  : parameters:
    - $ref:
  "#/components/parameters/IdSessionParam" - $ref:
  "#/components/parameters/IdMemberParam" get:
    summary: Logout a Member
    tags:
      - session management
    responses:
      '200':
        description: Logout member
        content:
          application/json:
            schema:
              type: array
              items:
                $ref: "#/components/schemas/Member"
      '204':
          description: No content
/sessions/{idSession}/stories:
  parameters:
    - $ref:
  "#/components/parameters/IdSessionParam" get:
    summary: list of user stories
    tags:
      - user stories
    responses:
      '200':
        description: The Resource
        content:
          application/json:
            schema:
              type: array
              items:
                $ref: "#/components/schemas/Member"
      '404':
          description: Not found
  post:
    summary: Creation of a user story
    requestBody:
      content:
        application/json:
          schema:
            $ref: "#/components/schemas/Member"
    tags:
      - user stories
    responses:
      '201': # status code
        description: Created JSON object
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/Member"
      '404':
        description: Not found

/sessions/{idSession}/stories/{idUserStory}
  : parameters:
    - $ref: "#/components/parameters/IdSessionParam"
  - $ref: "#/components/parameters/IdUserStoryParam"
  put:
    summary: Update story information
    requestBody:
      content:
        application/json:
          schema:
            $ref: "#/components/schemas/Member"
    tags:
      - user stories
    responses:
      '200': # status code
```

```yaml
        description: Created JSON object
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/Member"
      '404':
          description: Not found
  delete:
    summary: Delete user story
    tags:
      - user stories
    responses:
      '200': # status code
        description: Created JSON object
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/Member"
      '403':
          description: Forbidden
      '404':
          description: Not found

/sessions/{idSession}/votes:
  parameters:
    - $ref:
  "#/components/parameters/IdSessionParam" get:
    summary: List of votes emmited
    tags:
      - votations
    responses:
      '200':
        description: List of votes
        content:
          application/json:
            schema:
              type: array
              items:
                $ref: "#/components/schemas/Vote"
  post:
    summary: Emit a vote
    tags:
      - votations
    requestBody:
      content:
        application/json:
          schema:
              $ref: "#/components/schemas/Vote"
      responses:
        '201': # status code
          description: Created JSON object
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/Vote"
        '400':
            description: Bad Request
        '404':
            description: Not found
components:
  parameters:
    IdSessionParam:
      name: idSession
      in: path
      required: true
      schema:
        type: string
    IdMemberParam:
```

```yaml
      name: idMember
      in: path
      required: true
      schema:
        type: string
    IdUserStoryParam:
      name: idUserStory
      in: path
      required: true
      schema:
        type: string
  schemas:
    PokerPlanningSession
      : type: object
      required:
        - idSession
      properties:
        idSession:
          type: string
          readOnly: true
        title:
          type: string
    Member:
      type: object
      required:
        - idMember
        - name
      properties:
        idMember:
          type: string
          readOnly: true
        name:
          type: string
    UserStory:
      type: object
      properties:
        idUserStory:
          type: string
        description:
          type: string
        status:
          type: string
          enum:
            - PENDING
            - VOTING
            - VOTE
          default: PENDING
    Vote:
      type: object
      required:
        - idMember
        - idUserStory
      properties:
        idMember:
          type: string
        idUserStory:
          type: string
        value:
          type: string
```