Documentation

PDF Management and Collaboration Application

Application url: spot-draft-task-frontend.vercel.app

Frontend Repo: **https://github.com/mirirtiqa/SpotDraftTaskFrontend**
Backend Repo: **https://github.com/mirirtiqa/SpotDraftTaskBackend**
Backend Deployed on **Render**, server - **https://spotdrafttaskbackend.onrender.com**

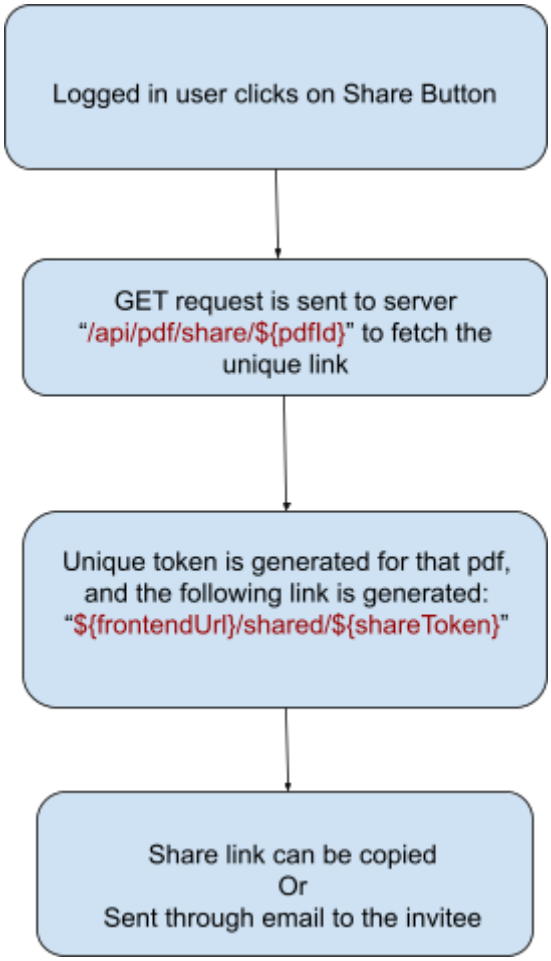
Features:

- Login/Signup with JWT-based auth
- Upload PDFs (only PDFs)
- Store PDFs in Google Cloud Storage
- View PDFs (secure access)
- Share PDFs using unique link or invite users using email
- Comment on PDFs (for both invited but authenticated and authenticated users)
- Dashboard that lists PDFs with search and date filter
- Store user information (name, hashed password, user's uploaded pdf details, comments on pdf's) in Mongodb
- Reset Password using link sent on email if user forgets password.
- Uses ContextAPI and LocalStorage to store and access user's name, email and token globally
- Responsive Website

Tech Stack:

- Next.js
- Material UI
- Context API
- Google Cloud Storage (via backend API)
- Mongodb
- Express
- Nodejs
- Nodemailer for sending emails

Sharing PDF Flow -



Viewing Shared PDFs and Comments from "{frontendUrl}/shared/${shareToken}" page

1. Request `${baseURL}/api/pdf/shared/${sharedToken}` is sent to server to fetch the pdf details
2. Server checks whether sharedToken is associated with any pdf.
3. If yes, it sends the PDF details

Viewing Shared PDFs comments

1. `${baseURL}/api/comments/get/${sharedToken}/${pdfId}` request is made
2. If token is present, then comments associated with the same pdf are sent

Adding Comments on Shared PDFs:

1. Post request `${baseURL}/api/comments/add/onshared` is made where body includes sharedToken field.
2. On backend, first check is performed whether sharedToken is associated with any PDF or not
3. If yes, the Comment Text included in the requests Body is added to the associated PDF

API Details

APIs for uploading, viewing, sharing PDFs and managing comments with and without authentication

```
server:
  - url: https://spotdrafttaskbackend.onrender.com

components:
  securitySchemes:
    bearerAuth:
      type: http
      scheme: bearer
      bearerFormat: JWT

  schemas:
    User:
      type: object
      properties:
        _id:
          type: string
        name:
          type: string
        email:
          type: string
        password:
          type: string
        resetPasswordToken:
            type: string
        resetPasswordExpires:
            type: Date
        pdfs:
          type: array
          items:
```

```
      type: object
      properties:
        pdfId:
          type: string
        name:
          type: string
        gcsFileName: //name of the file on google cloud storage
          type: string
        createdOn:
          type: string
          format: date-time
```

PDF:

```
  type: object
  properties:
    _id:
      type: string
    owner:
      type: string
    fileName:
      type: string
    gcsFileName:
      type: string
    sharedWith:
      type: array
      items:
        type: string
    createdAt:
      type: string
      format: date-time
    shareToken:
      type: string
```

Comment:

```
  type: object
  properties:
    _id:
      type: string
    pdfId:
      type: string
    authorName:
      type: string
    content:
```

```
      type: string
    createdAt:
      type: string
      format: date-time

LoginRequest:
  type: object
  required: [email, password]
  properties:
    email:
      type: string
    password:
      type: string

SignupRequest:
  type: object
  required: [name, email, password]
  properties:
    name:
      type: string
    email:
      type: string
    password:
      type: string

LoginResponse:
  type: object
  properties:
    token:
      type: string
    user:
      $ref: '#/components/schemas/User'

UploadResponse:
  type: object
  properties:
    message:
      type: string
    pdf:
      $ref: '#/components/schemas/PDF'

SharedLinkResponse:
  type: object
  properties:
```

```
        link:
          type: string

    SignedUrlResponse:
      type: object
      properties:
        url:
          type: string

paths:

  /api/auth/signup:
    post:
      summary: Register a new user
      tags: [Auth]
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/SignupRequest'
      responses:
        '201':
          description: User registered successfully
        '400':
          description: User already exists
        '500':
          description: Signup error

  /api/auth/login:
    post:
      summary: Login user
      tags: [Auth]
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/LoginRequest'
      responses:
        '200':
          description: Successful login
          content:
            application/json:
```

```
          schema:
            $ref: '#/components/schemas/LoginResponse'
      '400':
        description: Invalid email or password
      '500':
        description: Login error

/api/pdf/upload:
  post:
    summary: Upload a PDF file
    tags: [PDF]
    security: [{ bearerAuth: [] }]
    requestBody:
      required: true
      content:
        multipart/form-data:
          schema:
            type: object
            properties:
              pdf:
                type: string
                format: binary
    responses:
      '201':
        description: Uploaded successfully
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/UploadResponse'

/api/pdf/getpdfs:
  get:
    summary: List user's PDFs
    tags: [PDF]
    security: [{ bearerAuth: [] }]
    responses:
      '200':
        description: List of PDFs
        content:
          application/json:
            schema:
              type: array
              items:
                $ref: '#/components/schemas/PDF'
```

```
/api/pdf/share/{pdfId}:
  get:
    summary: Generate share link
    tags: [PDF]
    security: [{ bearerAuth: [] }]
    parameters:
      - in: path
        name: pdfId
        required: true
        schema:
          type: string
    responses:
      '200':
        description: Share link created
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/SharedLinkResponse'

/api/pdf/shared/{token}:
  get:
    summary: View shared PDF by token
    tags: [PDF]
    parameters:
      - in: path
        name: token
        required: true
        schema:
          type: string
    responses:
      '200':
        description: PDF data
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/PDF'

/api/pdf/getpdf/{pdfId}:
  get:
    summary: Get PDF by ID
    tags: [PDF]
    security: [{ bearerAuth: [] }]
    parameters:
```

```
      - in: path
        name: pdfId
        required: true
        schema:
          type: string
    responses:
      '200':
        description: PDF metadata
        content:
          application/json:
            schema:
              type: object
              properties:
                gcsFileName:
                  type: string
                fileName:
                  type: string
                pdfId:
                  type: string

/api/pdf/getpdfurl/{gcsfilename}:
  get:
    summary: Get signed URL for a PDF
    tags: [PDF]
    security: [{ bearerAuth: [] }]
    parameters:
      - in: path
        name: gcsfilename
        required: true
        schema:
          type: string
    responses:
      '200':
        description: Signed URL
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/SignedUrlResponse'

/api/pdf/getsharedpdfurl/{token}/{gcsfilename}:
  get:
    summary: Get signed URL for shared PDF
    tags: [PDF]
    parameters:
```

```
      - in: path
        name: token
        required: true
        schema:
          type: string
      - in: path
        name: gcsfilename
        required: true
        schema:
          type: string
    responses:
      '200':
        description: Signed URL
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/SignedUrlResponse'

/api/comments/add:
  post:
    summary: Add a comment to a PDF
    tags: [Comments]
    security: [{ bearerAuth: [] }]
    requestBody:
      required: true
      content:
        application/json:
          schema:
            type: object
            properties:
              authorName:
                type: string
              content:
                type: string
              pdfId:
                type: string
    responses:
      '201':
        description: Comment added
        content:
          application/json:
            schema:
              type: object
              properties:
```

```
            comment:
              $ref: '#/components/schemas/Comment'

/api/comments/add/onshared:
  post:
    summary: Add comment to shared PDF
    tags: [Comments]
    requestBody:
      required: true
      content:
        application/json:
          schema:
            type: object
            properties:
              authorName:
                type: string
              content:
                type: string
              pdfId:
                type: string
              sharedToken:
                type: string
    responses:
      '201':
        description: Comment added
        content:
          application/json:
            schema:
              type: object
              properties:
                comment:
                  $ref: '#/components/schemas/Comment'

/api/comments/get/{pdfId}:
  get:
    summary: Get comments for a PDF
    tags: [Comments]
    security: [{ bearerAuth: [] }]
    parameters:
      - in: path
        name: pdfId
        required: true
        schema:
          type: string
```

```
    responses:
      '200':
        description: List of comments
        content:
          application/json:
            schema:
              type: object
              properties:
                comments:
                  type: array
                  items:
                    $ref: '#/components/schemas/Comment'

/api/comments/get/{token}/{pdfId}:
  get:
    summary: Get comments for shared PDF
    tags: [Comments]
    parameters:
      - in: path
        name: token
        required: true
        schema:
          type: string
      - in: path
        name: pdfId
        required: true
        schema:
          type: string
    responses:
      '200':
        description: List of comments
        content:
          application/json:
            schema:
              type: object
              properties:
                comments:
                  type: array
                  items:
                    $ref: '#/components/schemas/Comment'

/api/auth/forgot-password:
  post:
```

```
    summary: Initiate password reset process by sending a reset link to the user's email by
embedding unique token in the email associated with the user “resetPasswordToken”
    tags:
      - Auth
    requestBody:
      required: true
      content:
        application/json:
          schema:
            type: object
            required:
              - email
            properties:
              email:
                type: string
                format: email
                example: "user@example.com"
    responses:
      '200':
        description: Password reset email sent
        content:
          application/json:
            schema:
              type: object
              properties:
                message:
                  type: string
                  example: "Password reset email sent"
      '404':
        description: User not found
        content:
          application/json:
            schema:
              type: object
              properties:
                message:
                  type: string
                  example: "User not found"
      '500':
        description: Error sending reset email
        content:
          application/json:
            schema:
              type: object
```

```
            properties:
              message:
                type: string
                example: "Error sending reset email"
              error:
                type: string
                example: "Detailed error message"

/api/auth/reset-password/{token}:
  post:
    summary: Reset user's password using a token from email
    tags:
      - Auth
    parameters:
      - in: path
        name: token
        required: true
        schema:
          type: string
        description: Password reset token
    requestBody:
      required: true
      content:
        application/json:
          schema:
            type: object
            required:
              - password
            properties:
              password:
                type: string
                format: password
                example: "newStrongPassword123"
    responses:
      '200':
        description: Password updated successfully
        content:
          application/json:
            schema:
              type: object
              properties:
                message:
                  type: string
                  example: "Password updated successfully"
```

```
'400':
  description: Invalid or expired token
  content:
    application/json:
      schema:
        type: object
        properties:
          message:
            type: string
            example: "Invalid or expired token"
'500':
  description: Error resetting password
  content:
    application/json:
      schema:
        type: object
        properties:
          message:
            type: string
          error:
            type: string
```