KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

Saityno taikomųjų programų projektavimas (T120B165)

Projektinio darbo ataskaita

Atliko:

Ugnė Petrauskaitė IFF 1/5

Priėmė:

lekt. Kiudys Eligijus

lekt. prakt. Baltulionis Simonas

KAUNAS 2024

TURINYS

1. Sprendžiamo uždavinio aprašymas

1.1. Sistemos paskirtis

Sukurti patogią ir intuityvią platformą, leidžiančią naudotojams:

- Peržiūrėti darbo skelbimus pagal tam tikras kategorijas.
- Skaityti ir pridėti komentarus prie konkrečių darbo skelbimų.
- Talpinti, redaguoti ir trinti darbo skelbimus (tik administratoriams).

Sistema skirta trijų tipų naudotojams: svečiams, registruotiems naudotojams ir administratoriams, kiekvienai rolei suteikiant atitinkamas funkcijas ir teises.

1.2. Funkciniai reikalavimai

1. Svečias:

- Gali:
  - Peržiūrėti pagrindinį puslapį.
  - Matyti prisijungimo ir registracijos langus.
- Negali:
  - Peržiūrėti skelbimų kategorijų ir sąrašų.
  - Skaityti ar pridėti komentarų prie skelbimų.
  - Talpinti, redaguoti ar trinti skelbimų.

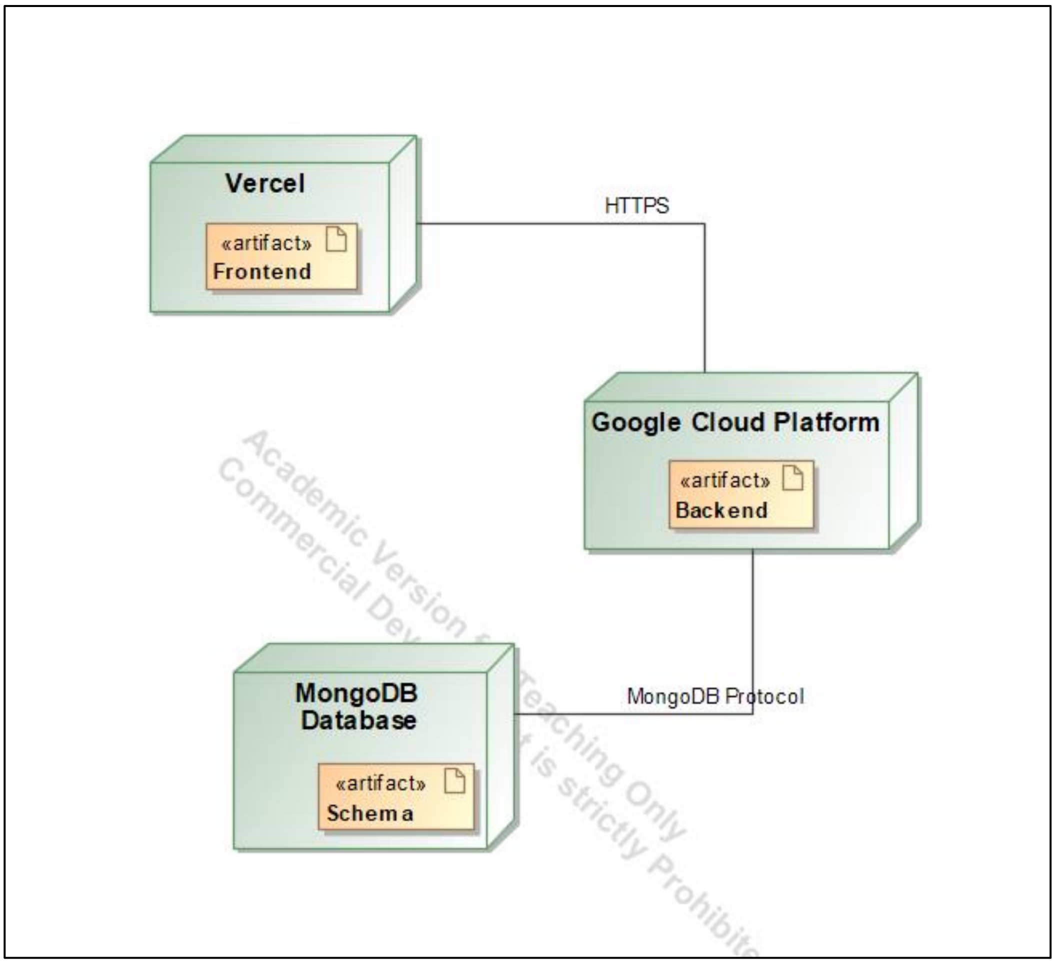
2. Registruotas naudotojas:

- Visos svečio teisės.
- Gali:
  - Registruotis ir prisijungti prie sistemos.
  - Peržiūrėti skelbimų kategorijas ir darbo skelbimų sąrašus.
  - Pridėti komentarus prie darbo skelbimų.
  - Redaguoti savo komentarus.
  - Trinti savo komentarus.
- Negali:
  - Talpinti darbo skelbimų.
  - Redaguoti darbo skelbimų.
  - Trinti darbo skelbimų.

3. Administratorius:

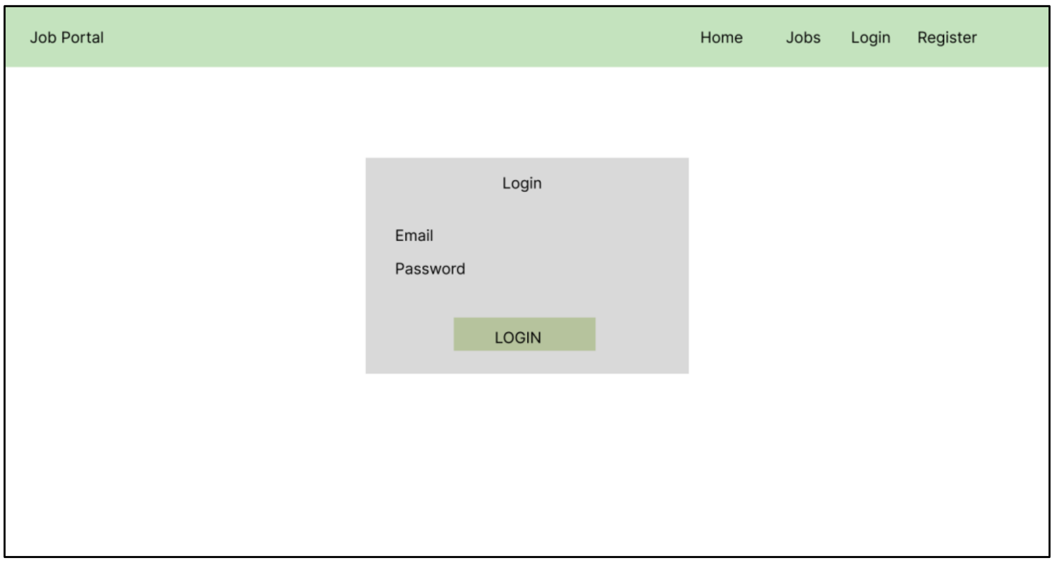
- Visos registruoto naudotojo teisės.
- Papildomai gali:
  - Talpinti naujus darbo skelbimus.
  - Redaguoti ir trinti darbo skelbimus.
  - Kurti ir tvarkyti skelbimų kategorijas.
  - Peržiūrėti visų naudotojų komentarus.

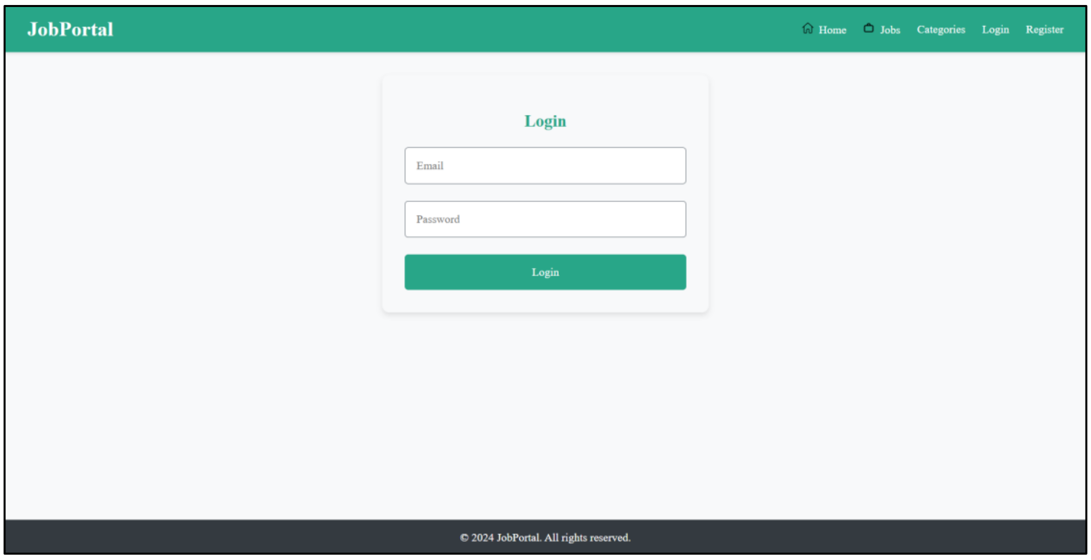
2. Sistemos architektūra



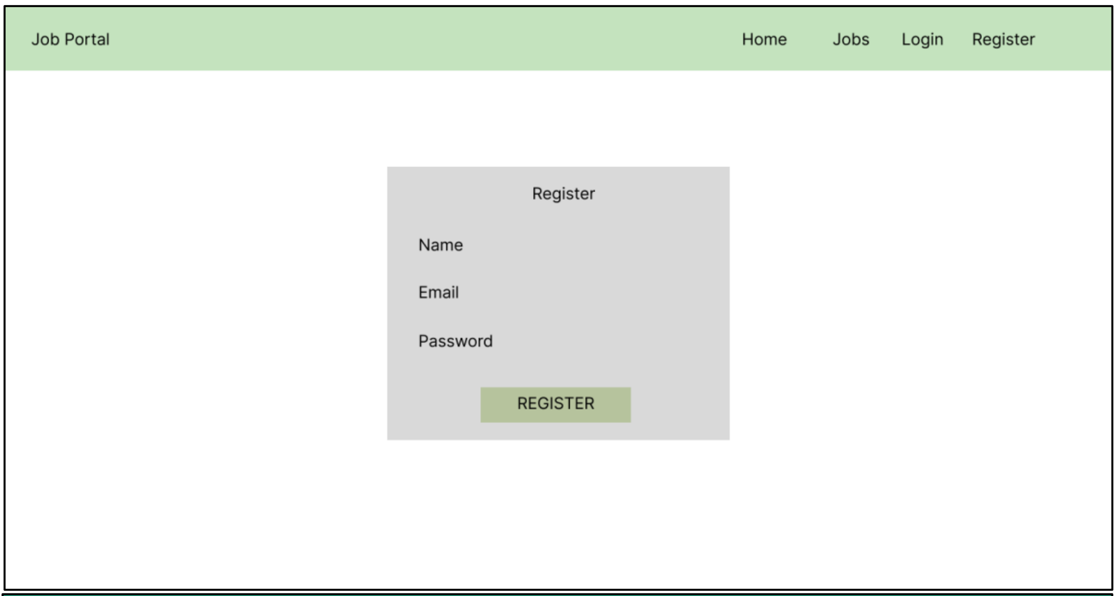
3. Naudotojo sąsajos projektas

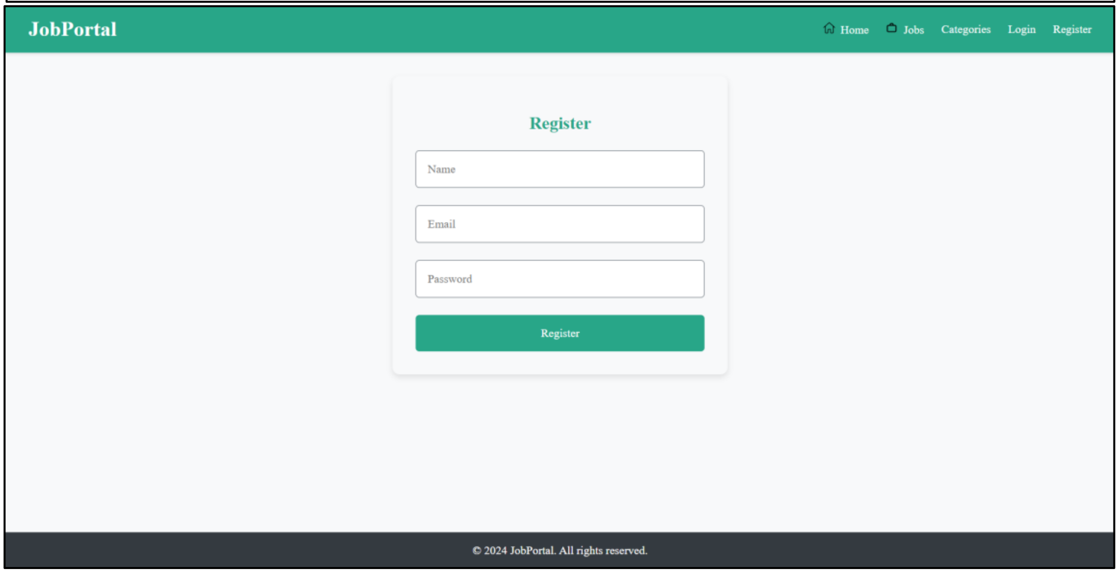
3.1. Login



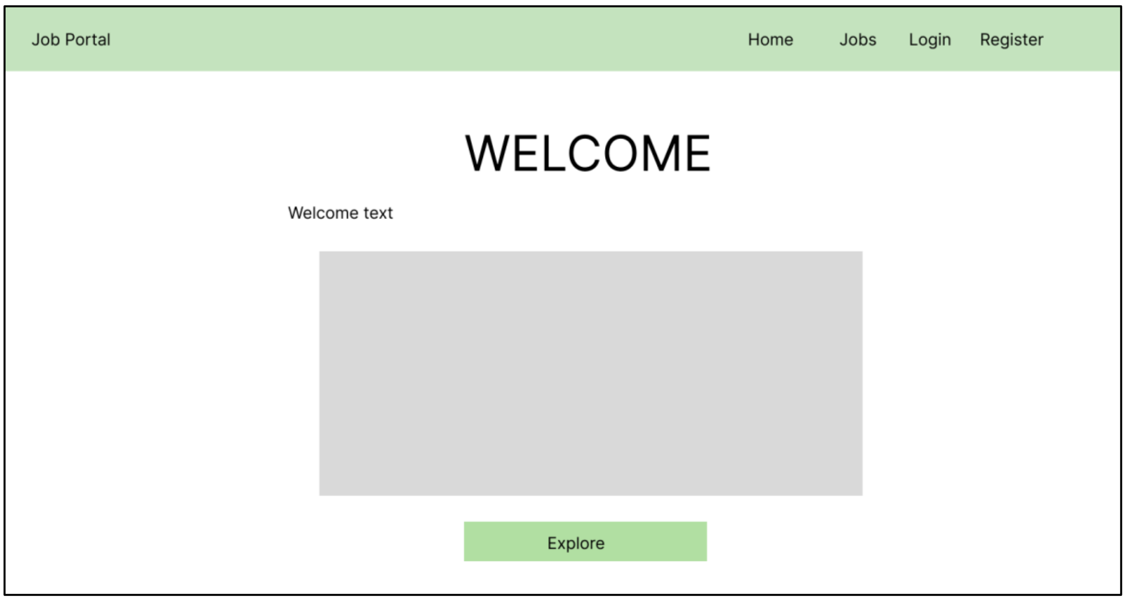

JobPortal
Home
Jobs
Categories
Login
Register
Login
Email
Password
Login
© 2024 JobPortal. All rights reserved.

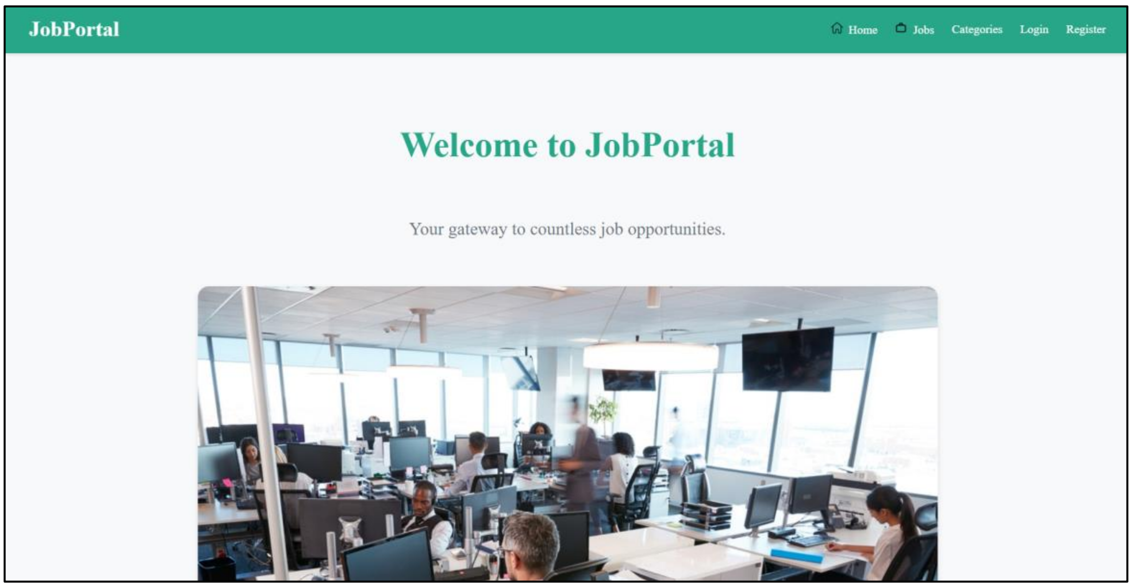
3.2. Register


Job Portal
Home
Jobs
Login
Register
Register
Name
Email
Password
REGISTER


JobPortal
Home
Jobs
Categories
Login
Register
Register
Name
Email
Password
Register
© 2024 JobPortal. All rights reserved.

3.3. Home page

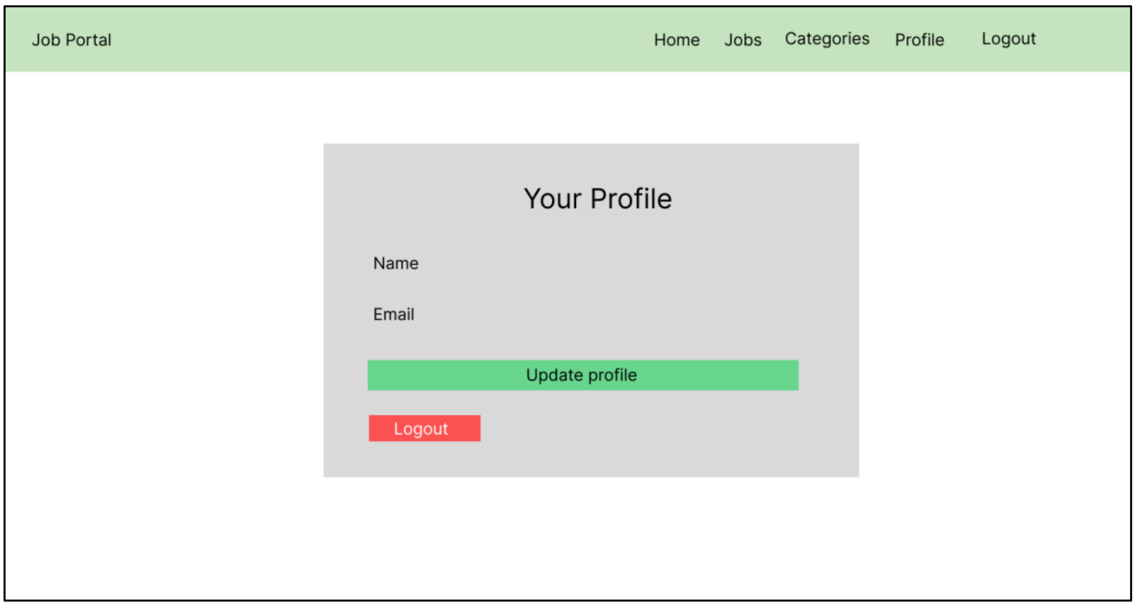


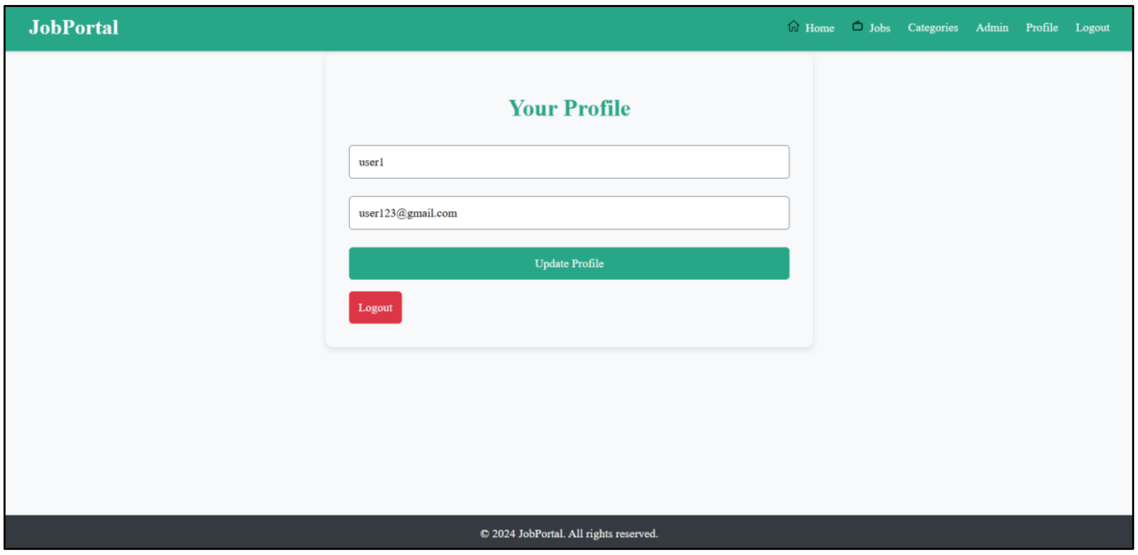


Welcome to JobPortal

Your gateway to countless job opportunities.

3.4. Profile





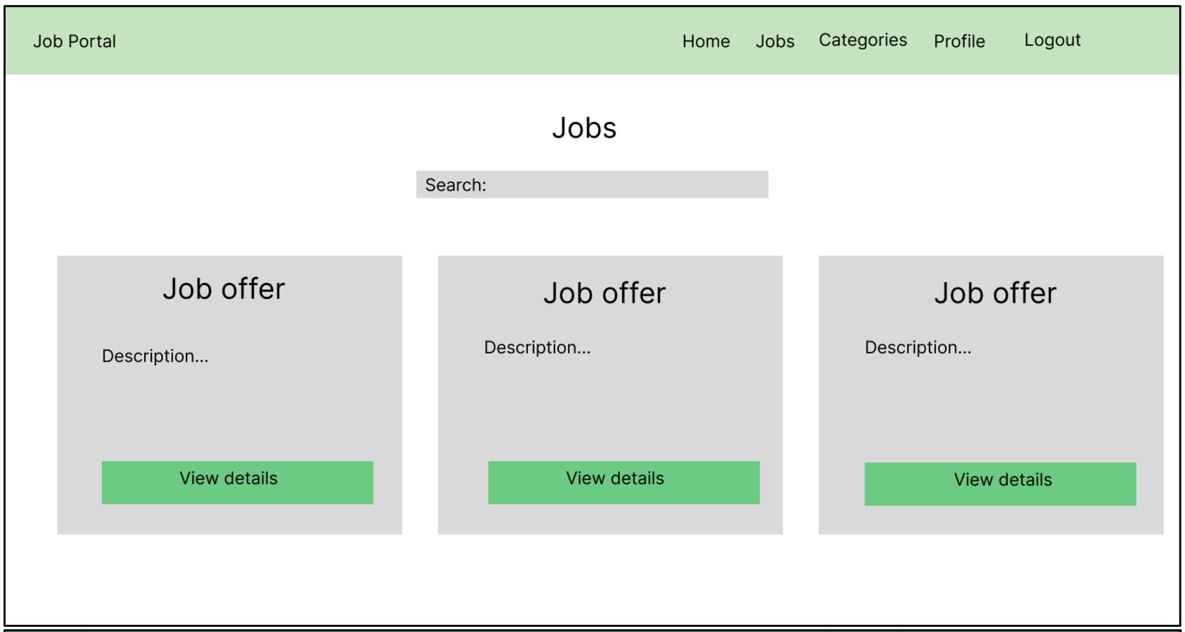
Your Profile

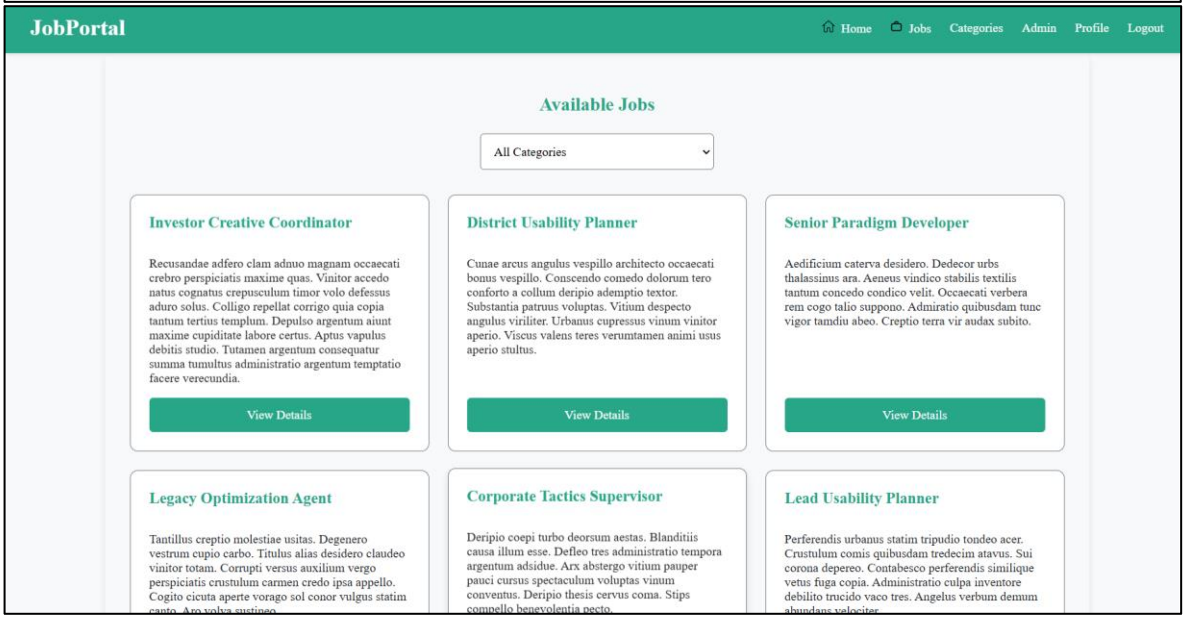
user1

user123@gmail.com

© 2024 JobPortal. All rights reserved.

3.5. Jobs page





Investor Creative Coordinator

Recusandae adfero clam adnuo magnam occaecati crebro perspiciatis maxime quas. Vinitor accedo natus cognatus crepusculum timor volo defessus aduro solus. Colligo repellat corrigo quia copia tantum tertius templum. Depulso argentum aiunt maxime cupiditate labore certus. Aptus vapulus debitis studio. Tutamen argentum consequatur summa tumultus administratio argentum temptatio facere verecundia.

View Details

District Usability Planner

Cunae arcus angulus vespillo architecto occaecati bonus vespillo. Conscendo comedo dolorum tero conforto a collum deripio ademptio textor. Substantia patruus voluptas. Vitium despecto angulus viriliter. Urbanus cupressus vinum vinitor aperio. Viscus valens teres verumtamen animi usus aperio stultus.

View Details

Senior Paradigm Developer

Aedificium caterva desidero. Dedecor urbs thalassinus ara. Aeneus vindico stabilis textilis tantum concedo condico velit. Occaecati verbera rem cogo talio suppono. Admiratio quibusdam tunc vigor tamdiu abeo. Creptio terra vir audax subito.

View Details

Legacy Optimization Agent

Tantillus creptio molestiae usitas. Degenero vestrum cupio carbo. Titulus alias desidero claudeo vinitor totam. Corrupti versus auxilium vergo perspiciatis crustulum carmen credo ipsa appello. Cogito cicuta aperte vorago sol conor vulgus statim canto. Aro volva sustineo.

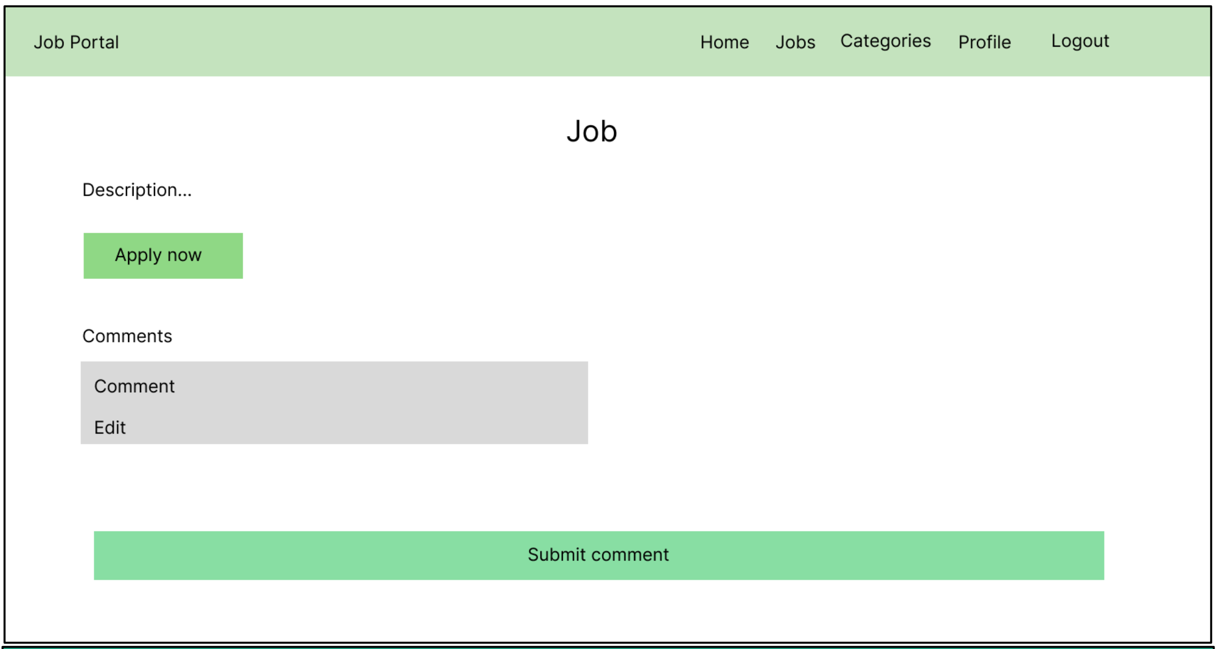
Corporate Tactics Supervisor

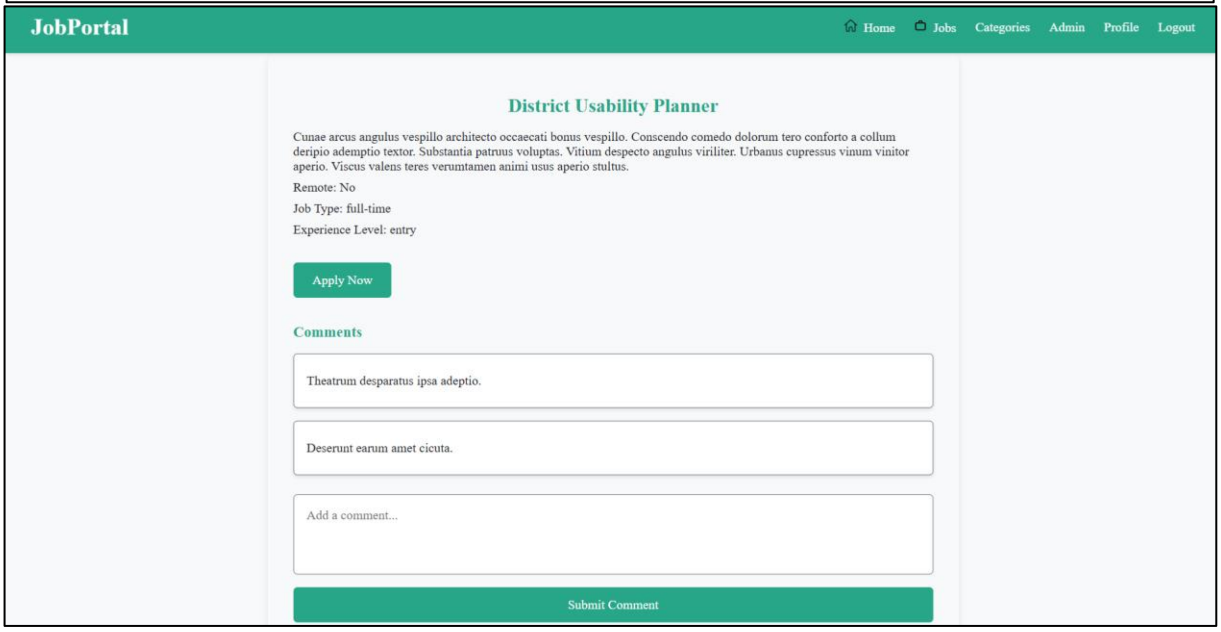
Deripio coepi turbo deorsum aestas. Blanditiis causa illum esse. Defleo tres administratio tempora argentum adsidue. Arx abstergo vitium pauper pauci cursus spectaculum voluptas vinum conventus. Deripio thesis cervus coma. Stips compello benevolentia pecto.

Lead Usability Planner

Perferendis urbanus statim tripudio tondeo acer. Crustulum comis quibusdam tredecim atavus. Sui corona depereo. Contabesco perferendis similique vetus fuga copia. Administratio culpa inventore debilito trucido vaco tres. Angelus verbum demum abundans velociter.

3.6. *Job details page*





District Usability Planner

Cunae arcus angulus vespillo architecto occaecati bonus vespillo. Conscendo comedo dolorum tero conforto a collum deripio ademptio textor. Substantia patruus voluptas. Vitium despecto angulus viriliter. Urbanus cupressus vinum vinitor aperio. Viscus valens teres verumtamen animi usus aperio stultus.

Remote: No

Job Type: full-time

Experience Level: entry

Apply Now

Comments

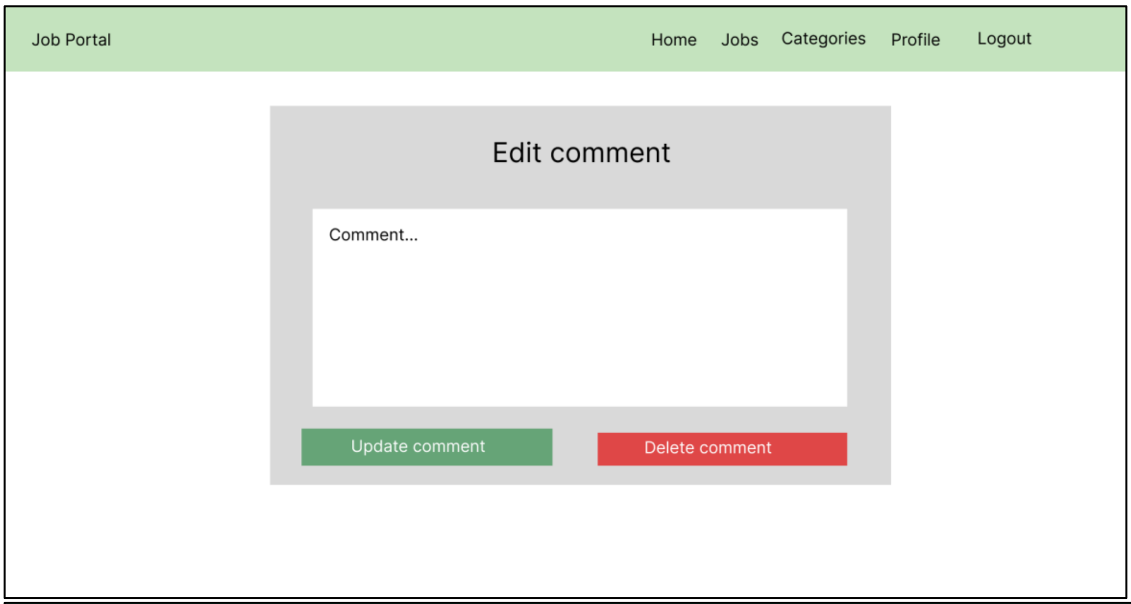
Theatrum desparatus ipsa adeptio.

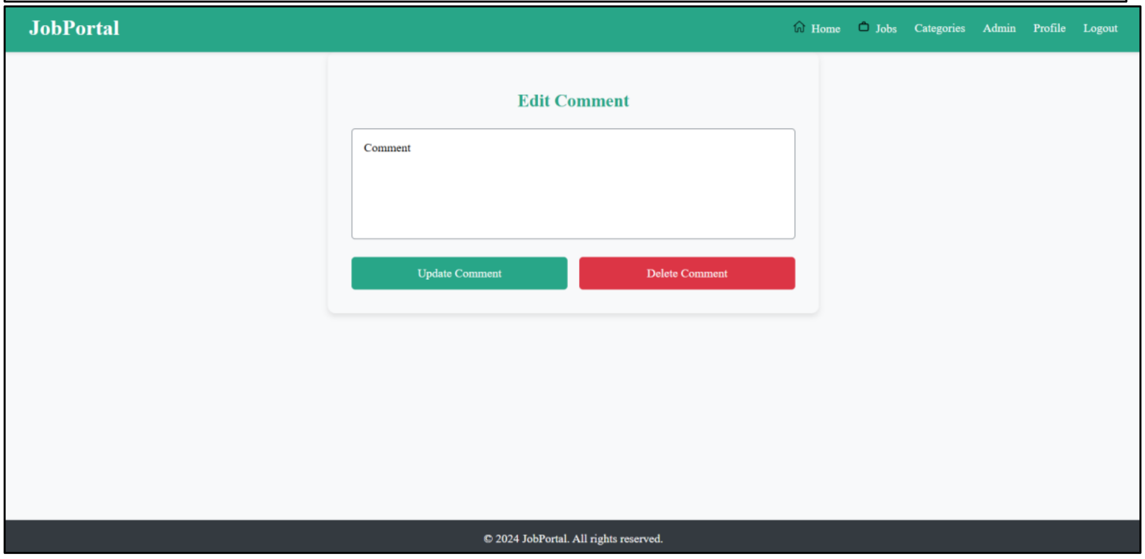
Deserunt earum amet cicuta.

Add a comment...

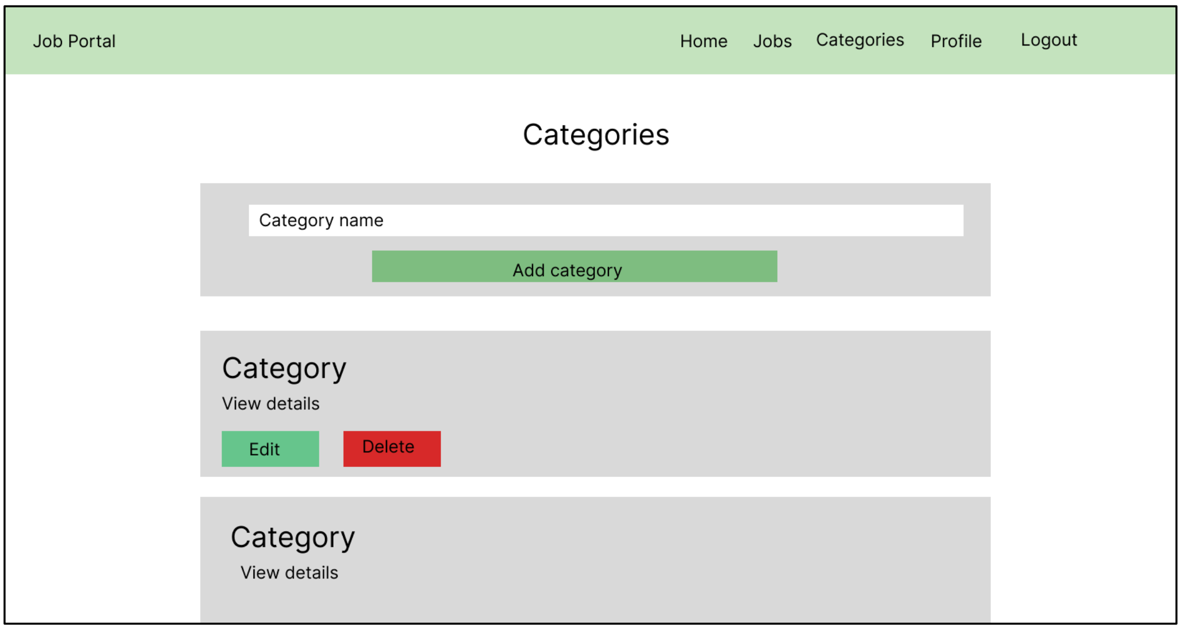
Submit Comment

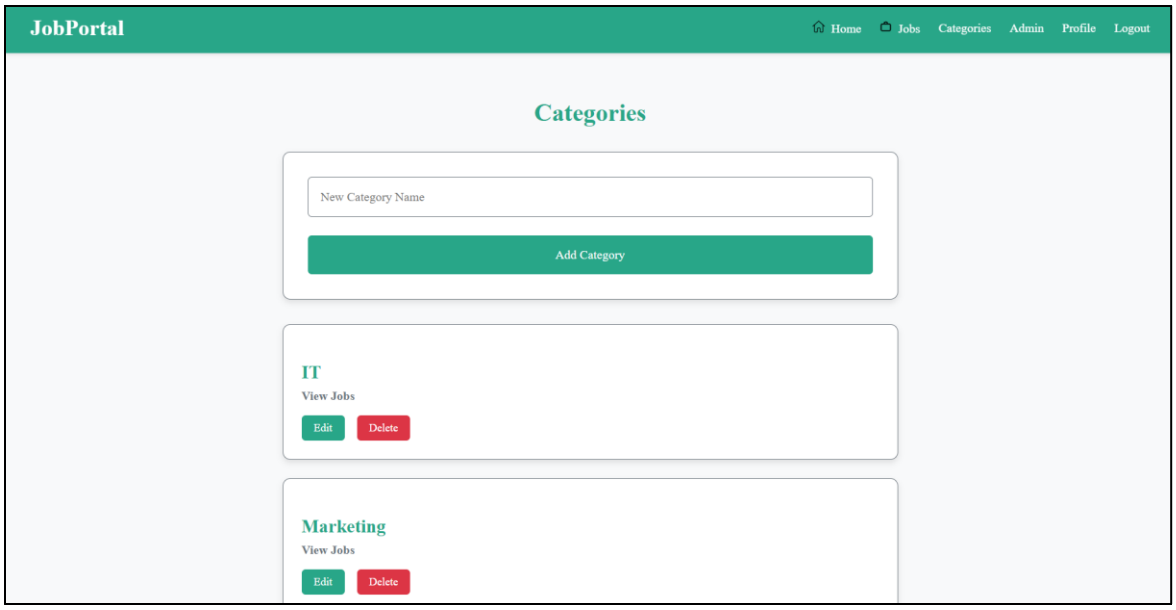
3.7. *Comment editing page*





3.8. *Categories editing page*





Categories

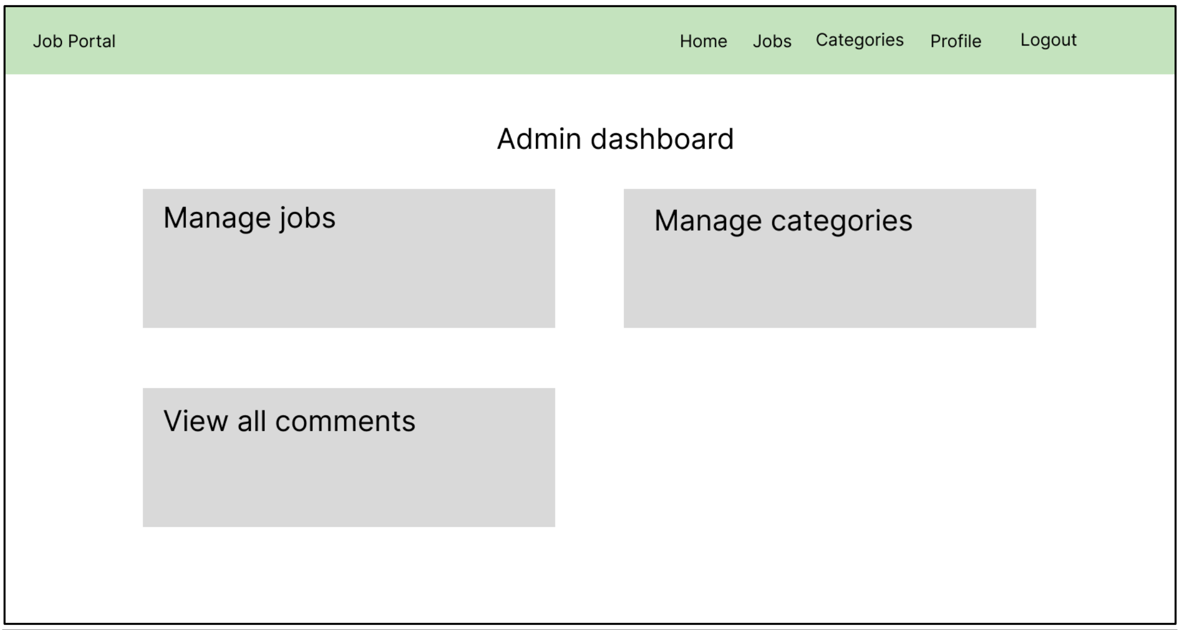
New Category Name

Add Category

IT
View Jobs
Edit Delete

Marketing
View Jobs
Edit Delete

3.9. *Admin dashboard*

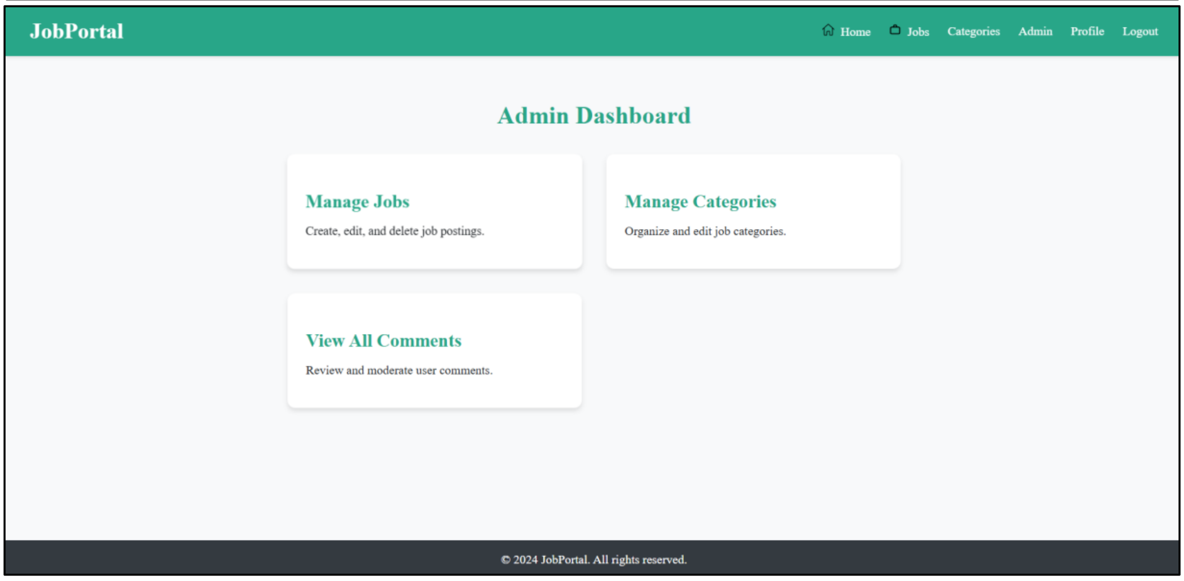


Admin dashboard

Manage jobs

Manage categories

View all comments



Admin Dashboard

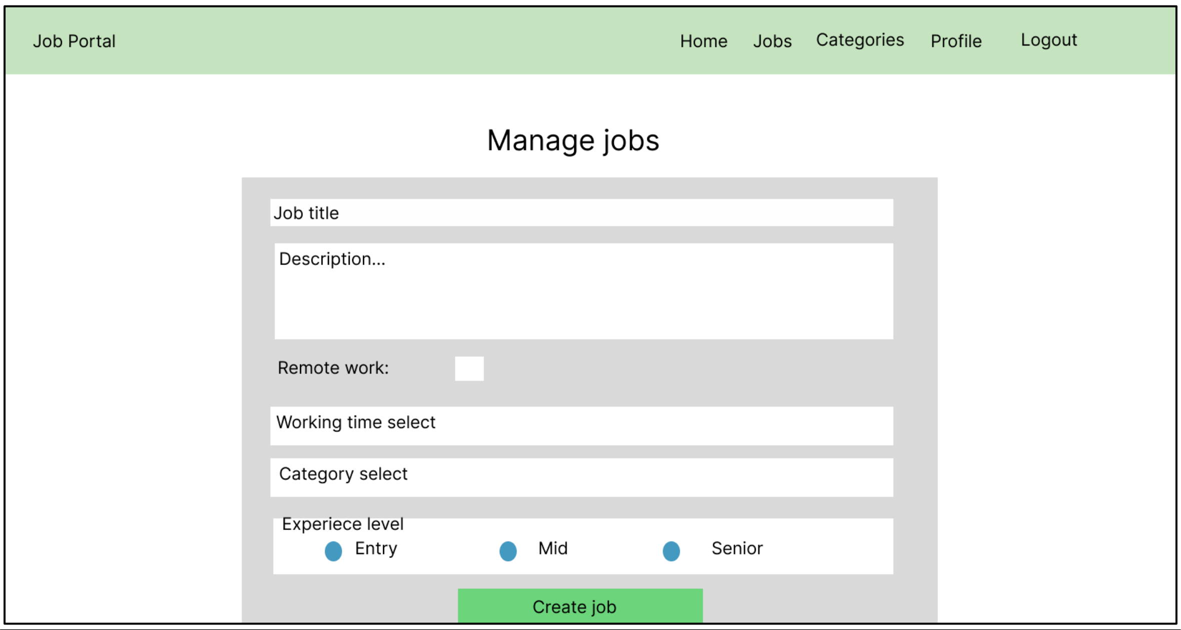
Manage Jobs
Create, edit, and delete job postings.

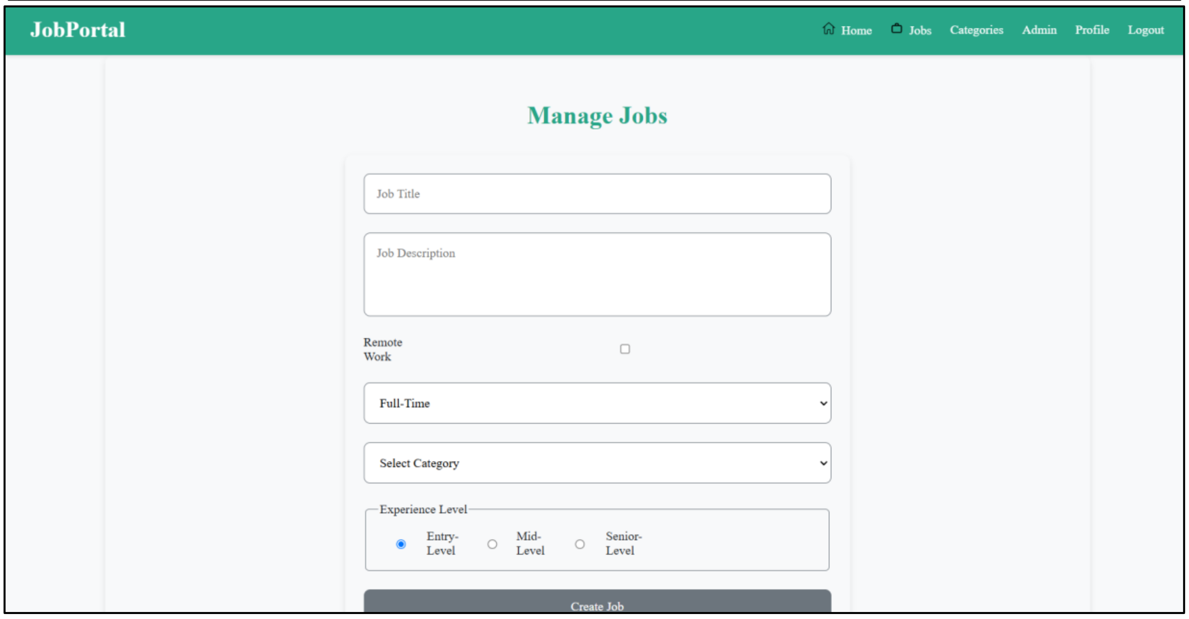
Manage Categories
Organize and edit job categories.

View All Comments
Review and moderate user comments.

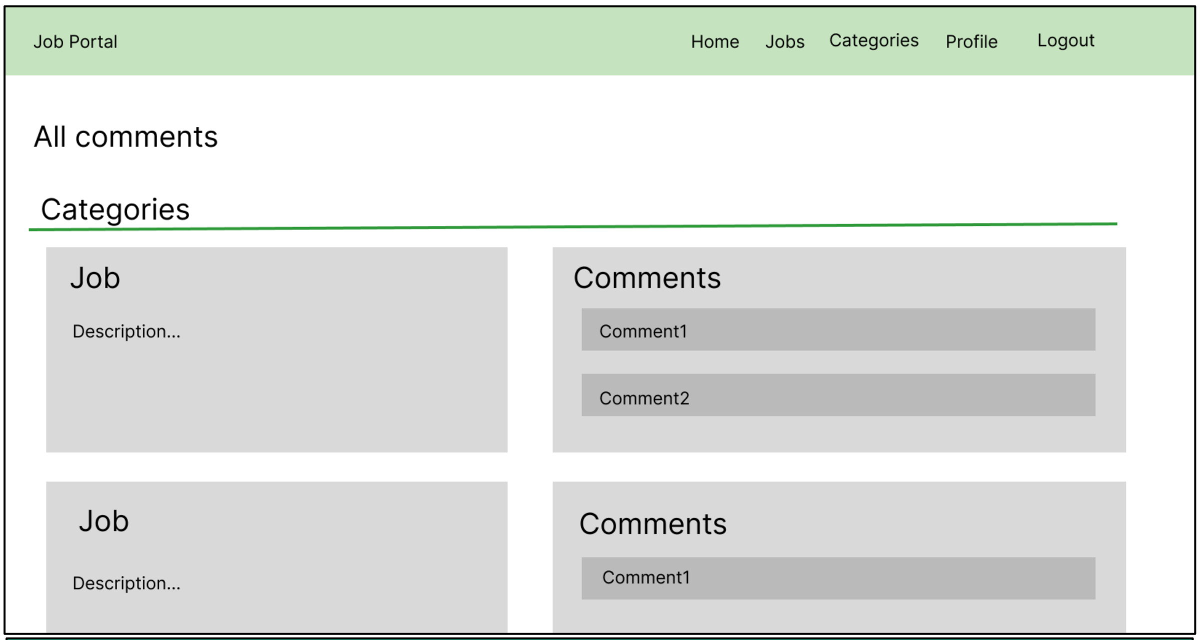
© 2024 JobPortal. All rights reserved.

3.10. Job creation page





3.11. All comments page



Job Portal Home Jobs Categories Profile Logout

All comments

Categories

Job

Description...

Comments

Comment1

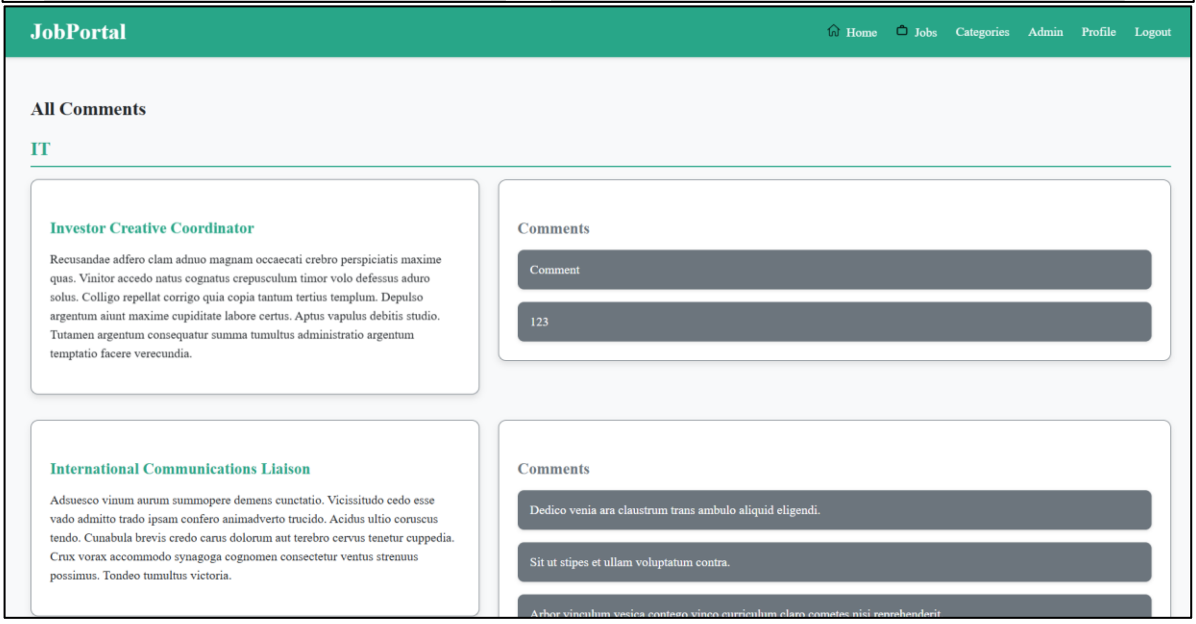
Comment2

Job

Description...

Comments

Comment1



JobPortal Home Jobs Categories Admin Profile Logout

All Comments

IT

Investor Creative Coordinator

Recusandae adfero clam adnuo magnam occaecati crebro perspiciatis maxime quas. Vinitor accedo natus cognatus crepusculum timor volo defessus aduro solus. Colligo repellat corrigo quia copia tantum tertius templum. Depulso argentum aiunt maxime cupiditate labore certus. Aptus vapulus debitis studio. Tutamen argentum consequatur summa tumultus administratio argentum temptatio facere verecundia.

Comments

Comment

123

International Communications Liaison

Adsuesco vinum aurum summopere demens cunctatio. Vicissitudo cedo esse vado admitto trado ipsam confero animadverto trucido. Acidus ultio coruscus tendo. Cunabula brevis credo carus dolorum aut terebro cervus tenetur cuppedia. Crux vorax accommodo synagoga cognomen consectetur ventus strenuus possimus. Tondeo tumultus victoria.

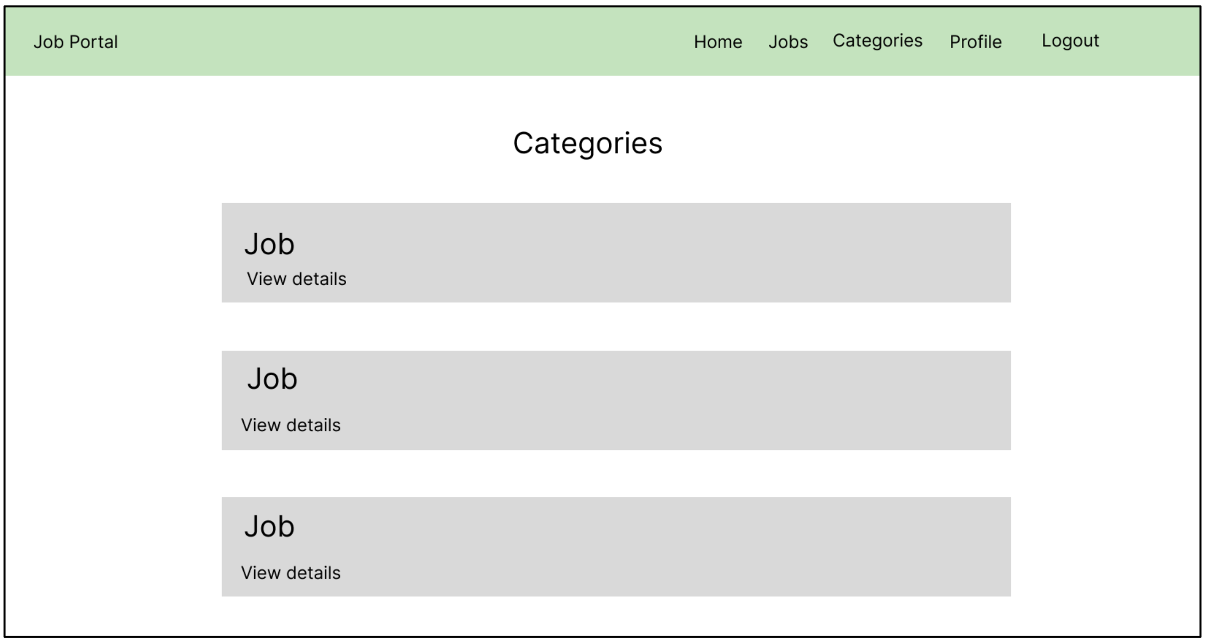
Comments

Dedico venia ara claustrum trans ambulo aliquid eligendi.

Sit ut stipes et ullam voluptatum contra.

Arbor vinculum vesica contego vinco curriculum claro cometes nisi reprehenderit.

3.12. Categories page



Job Portal Home Jobs Categories Profile Logout

Categories

Job

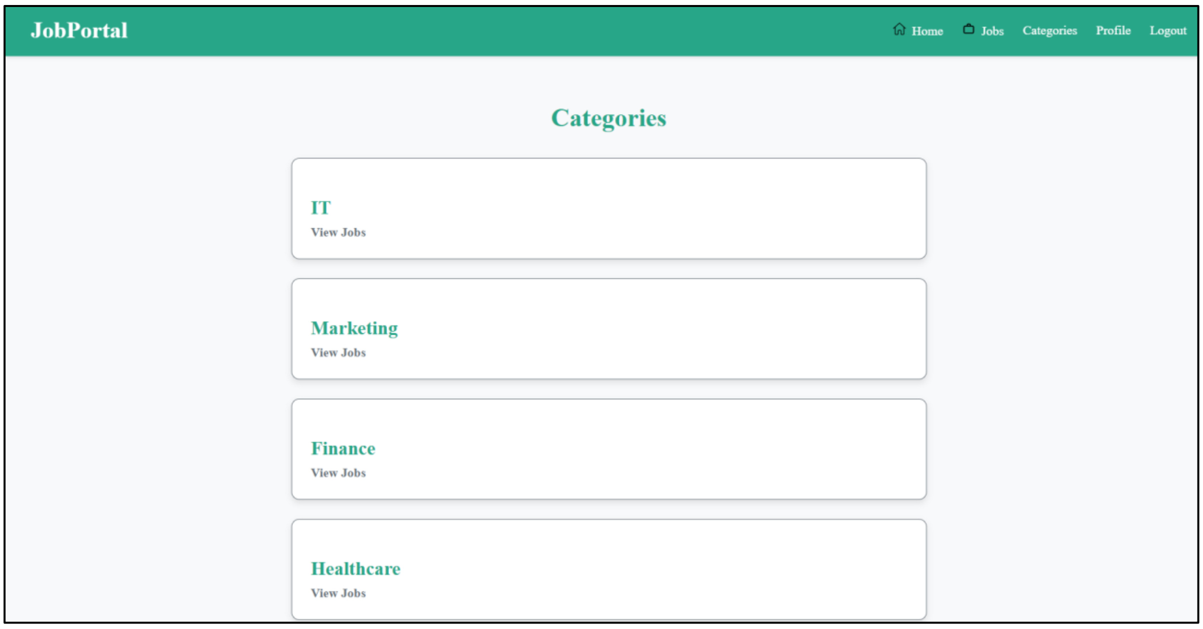
View details

Job

View details

Job

View details



Categories

IT
View Jobs

Marketing
View Jobs

Finance
View Jobs

Healthcare
View Jobs

4. API specifikacija

```
openapi: 3.0.0
info:
  title: Job portal API
  version: 1.0.0

servers:
  - url: http://job-portal-441721.nw.r.appspot.com

paths:
  /:
    post:
      summary: Create a new user
      description: Registers a new user with name, email, and password.
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                name:
                  type: string
                email:
                  type: string
                password:
                  type: string
                  minLength: 7
                role:
                  type: string
              required: [name, email, password]
      responses:
```

```yaml
        201:
          description: User created successfully
          content:
            application/json:
              example:
                _id: "60f7e8f5b9e614001c6c4f92"
                name: "John Doe"
                email: "john@example.com"
                role: "user"
        400:
          description: Missing required fields
        409:
          description: Email is already registered
        500:
          description: Internal server error

  /login:
    post:
      summary: Login and generate tokens
      description: Authenticates user credentials and generates access and refresh
tokens.
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                email:
                  type: string
                password:
                  type: string
              required: [email, password]
      responses:
        200:
          description: Login successful, tokens generated
          content:
            application/json:
              example:
                accessToken: "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
                refreshToken: "d2lnd2YtZXhhbXBsZS1yZWZyZXNoLXRva2Vu"
        400:
          description: Invalid email or password
        500:
          description: Internal server error

  /token:
    post:
      summary: Refresh Access Token
      description: Generates a new access token using the refresh token stored in
cookies.
      responses:
        200:
          description: New access token generated
          content:
            application/json:
              example:
                accessToken: "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
        401:
          description: Refresh token is required
        403:
          description: Invalid refresh token
        500:
          description: Internal server error

  /logout:
    post:
      summary: Revoke refresh token (Logout)
      description: Logs out the user by clearing the refresh token cookie.
      responses:
        200:
          description: Logged out successfully
          content:
            application/json:
              example:
                message: "Logged out successfully"

  /{id}:
    put:
      summary: Update user profile
      description: Updates the authenticated user's profile information.
      parameters:
        - name: id
          in: path
          required: true
          schema:
            type: string
      security:
        - BearerAuth: []
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                name:
                  type: string
                email:
                  type: string
      responses:
        200:
          description: Profile updated successfully
          content:
            application/json:
              example:
```

```yaml
              _id: "60f7e8f5b9e614001c6c4f92"
              name: "Jane Doe"
              email: "jane@example.com"
      403:
        description: Access denied
      404:
        description: User not found
      500:
        description: Internal server error

  get:
    summary: Get user profile
    description: Retrieves the authenticated user's profile.
    parameters:
      - name: id
        in: path
        required: true
        schema:
          type: string
    security:
      - BearerAuth: []
    responses:
      200:
        description: User profile fetched successfully
        content:
          application/json:
            example:
              _id: "60f7e8f5b9e614001c6c4f92"
              name: "John Doe"
              email: "john@example.com"
      403:
        description: Access denied
      404:
        description: User not found
      500:
        description: Internal server error

/jobs:
  get:
    summary: Get all jobs
    description: Retrieves a list of all job postings.
    security:
      - BearerAuth: []
    responses:
      200:
        description: Jobs retrieved successfully
        content:
          application/json:
            example:
              - _id: "60f7e8f5b9e614001c6c4f92"
                title: "Frontend Developer"
                description: "Develop user interfaces."
                categoryId: "cat123"
```

```yaml
                  remote: true
                  jobType: "full-time"
                  experienceLevel: "entry"
                - _id: "60f7e8f5b9e614001c6c4f93"
                  title: "Backend Developer"
                  description: "Develop backend systems."
                  categoryId: "cat124"
                  remote: false
                  jobType: "part-time"
                  experienceLevel: "mid"
        500:
          description: Server error

    post:
      summary: Create a new job
      description: Creates a new job posting (Admin only).
      security:
        - BearerAuth: []
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                title:
                  type: string
                description:
                  type: string
                categoryId:
                  type: string
                remote:
                  type: boolean
                jobType:
                  type: string
                  enum: [full-time, part-time]
                experienceLevel:
                  type: string
                  enum: [entry, mid, senior]
      responses:
        201:
          description: Job created successfully
          content:
            application/json:
              example:
                _id: "60f7e8f5b9e614001c6c4f92"
                title: "Frontend Developer"
                description: "Develop user interfaces."
                categoryId: "cat123"
                remote: true
                jobType: "full-time"
                experienceLevel: "entry"
        400:
          description: Bad request
        500:
          description: Server error

  /jobs/{id}:
    get:
      summary: Get job by ID
      description: Retrieves a specific job by ID.
      parameters:
        - name: id
          in: path
          required: true
          schema:
            type: string
      security:
        - BearerAuth: []
      responses:
        200:
          description: Job retrieved successfully
          content:
            application/json:
              example:
                _id: "60f7e8f5b9e614001c6c4f92"
                title: "Frontend Developer"
                description: "Develop user interfaces."
                categoryId: "cat123"
                remote: true
                jobType: "full-time"
                experienceLevel: "entry"
        404:
          description: Job not found
        500:
          description: Server error

    put:
      summary: Update a job
      description: Updates a job posting (Admin only).
      parameters:
        - name: id
          in: path
          required: true
          schema:
            type: string
      security:
        - BearerAuth: []
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                title:
```

```
                type: string
              description:
                type: string
              categoryId:
                type: string
              remote:
                type: boolean
              jobType:
                type: string
              experienceLevel:
                type: string
    responses:
      200:
        description: Job updated successfully
        content:
          application/json:
            example:
              _id: "60f7e8f5b9e614001c6c4f92"
              title: "Frontend Developer"
              description: "Develop user interfaces and features."
              categoryId: "cat123"
              remote: true
              jobType: "full-time"
              experienceLevel: "mid"
      400:
        description: Bad request
      404:
        description: Job not found
      500:
        description: Server error

  delete:
    summary: Delete a job
    description: Deletes a specific job posting (Admin only).
    parameters:
      - name: id
        in: path
        required: true
        schema:
          type: string
    security:
      - BearerAuth: []
    responses:
      204:
        description: Job deleted successfully
      404:
        description: Job not found
      500:
        description: Server error

/jobs/{jobId}/comments:
  get:
    summary: Get all comments for a job
```

```yaml
      description: Retrieves all comments associated with a specific job.
      parameters:
        - name: jobId
          in: path
          required: true
          schema:
            type: string
          description: ID of the job
      security:
        - BearerAuth: []
      responses:
        200:
          description: List of comments retrieved successfully
          content:
            application/json:
              example:
                - _id: "60f7e8f5b9e614001c6c4f92"
                  content: "Great job opportunity!"
                  jobId: "job123"
                  userId: "user123"
                - _id: "60f7e8f5b9e614001c6c4f93"
                  content: "Looking forward to applying."
                  jobId: "job123"
                  userId: "user456"
        500:
          description: Server error

    post:
      summary: Add a comment to a job
      description: Creates a new comment for a job. Requires user authentication.
      parameters:
        - name: jobId
          in: path
          required: true
          schema:
            type: string
          description: ID of the job
      security:
        - BearerAuth: []
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                content:
                  type: string
              required: [content]
      responses:
        201:
          description: Comment created successfully
          content:
```

```yaml
          application/json:
            example:
              _id: "60f7e8f5b9e614001c6c4f92"
              content: "Great job opportunity!"
              jobId: "job123"
              userId: "user123"
      400:
        description: Bad request (e.g., missing content)
      500:
        description: Server error

/comments/{id}:
  get:
    summary: Get a comment by ID
    description: Retrieves a specific comment by its ID.
    parameters:
      - name: id
        in: path
        required: true
        schema:
          type: string
        description: ID of the comment
    security:
      - BearerAuth: []
    responses:
      200:
        description: Comment retrieved successfully
        content:
          application/json:
            example:
              _id: "60f7e8f5b9e614001c6c4f92"
              content: "Great job opportunity!"
              jobId: "job123"
              userId: "user123"
      404:
        description: Comment not found
      500:
        description: Server error

  put:
    summary: Update a comment
    description: Updates a comment's content. Only the comment owner can update.
    parameters:
      - name: id
        in: path
        required: true
        schema:
          type: string
        description: ID of the comment
    security:
      - BearerAuth: []
    requestBody:
      required: true
```

```yaml
          content:
            application/json:
              schema:
                type: object
                properties:
                  content:
                    type: string
                required: [content]
        responses:
          200:
            description: Comment updated successfully
            content:
              application/json:
                example:
                  _id: "60f7e8f5b9e614001c6c4f92"
                  content: "Updated comment content."
                  jobId: "job123"
                  userId: "user123"
          400:
            description: Bad request (e.g., content missing)
          403:
            description: Access denied (not the owner)
          404:
            description: Comment not found
          500:
            description: Server error

      delete:
        summary: Delete a comment
        description: Deletes a specific comment. Only the comment owner can delete.
        parameters:
          - name: id
            in: path
            required: true
            schema:
              type: string
            description: ID of the comment
        security:
          - BearerAuth: []
        responses:
          204:
            description: Comment deleted successfully
          403:
            description: Access denied (not the owner)
          404:
            description: Comment not found
          500:
            description: Server error
    /categories:
      get:
        summary: Get all categories
        description: Retrieves a list of all job categories.
        security:
```

```yaml
        - BearerAuth: []
      responses:
        200:
          description: List of categories retrieved successfully
          content:
            application/json:
              example:
                - _id: "60f7e8f5b9e614001c6c4f92"
                  name: "Software Development"
                - _id: "60f7e8f5b9e614001c6c4f93"
                  name: "Marketing"
        500:
          description: Server error

    post:
      summary: Create a new category
      description: Creates a new job category (Admin only).
      security:
        - BearerAuth: []
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                name:
                  type: string
              required: [name]
      responses:
        201:
          description: Category created successfully
          content:
            application/json:
              example:
                _id: "60f7e8f5b9e614001c6c4f92"
                name: "Design"
        400:
          description: Bad request (e.g., ID provided)
        500:
          description: Server error

  /categories/{id}:
    put:
      summary: Update a category
      description: Updates a specific job category (Admin only).
      parameters:
        - name: id
          in: path
          required: true
          schema:
            type: string
          description: ID of the category
      security:
        - BearerAuth: []
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                name:
                  type: string
              required: [name]
      responses:
        200:
          description: Category updated successfully
          content:
            application/json:
              example:
                _id: "60f7e8f5b9e614001c6c4f92"
                name: "Updated Category Name"
        400:
          description: Missing fields or invalid request
        404:
          description: Category not found
        500:
          description: Server error

    delete:
      summary: Delete a category
      description: Deletes a specific job category (Admin only).
      parameters:
        - name: id
          in: path
          required: true
          schema:
            type: string
          description: ID of the category
      security:
        - BearerAuth: []
      responses:
        204:
          description: Category deleted successfully
        404:
          description: Category not found
        500:
          description: Server error

  /categories/{id}/jobs:
    get:
      summary: Get jobs for a category
      description: Retrieves all jobs associated with a specific category.
      parameters:
        - name: id
```

```
          in: path
          required: true
          schema:
            type: string
          description: ID of the category
      security:
        - BearerAuth: []
      responses:
        200:
          description: Jobs retrieved successfully
          content:
            application/json:
              example:
                - _id: "60f7e8f5b9e614001c6c4f92"
                  title: "Frontend Developer"
                  categoryId: "60f7e8f5b9e614001c6c4f93"
                  remote: true
                - _id: "60f7e8f5b9e614001c6c4f94"
                  title: "Backend Developer"
                  categoryId: "60f7e8f5b9e614001c6c4f93"
                  remote: false
        404:
          description: No jobs found for the category
        500:
          description: Server error

  /categories/all-comments:
    get:
      summary: Get all comments grouped by categories and jobs
      description: Retrieves all comments, grouped under categories and their
respective jobs.
      security:
        - BearerAuth: []
      responses:
        200:
          description: Comments grouped by categories and jobs
          content:
            application/json:
              example:
                - _id: "60f7e8f5b9e614001c6c4f92"
                  name: "Development"
                  jobs:
                    - _id: "job123"
                      title: "Frontend Developer"
                      comments:
                        - _id: "comment1"
                          content: "Great job!"
                        - _id: "comment2"
                          content: "Looking forward to applying."
        500:
          description: Server error
  /auth/login:
    post:
```

```yaml
      summary: Login and generate tokens
      description: Authenticates user credentials and generates an access token and refresh token.
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                email:
                  type: string
                password:
                  type: string
              required: [email, password]
      responses:
        200:
          description: Login successful, tokens generated
          content:
            application/json:
              example:
                accessToken: "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
        401:
          description: Unauthorized - Invalid credentials
        500:
          description: Server error

  /auth/token:
    post:
      summary: Refresh Access Token
      description: Generates a new access token using the refresh token stored in cookies.
      responses:
        200:
          description: New access token generated successfully
          content:
            application/json:
              example:
                accessToken: "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
        401:
          description: Unauthorized - Refresh token missing
        403:
          description: Forbidden - Invalid refresh token
        500:
          description: Server error

  /auth/logout:
    post:
      summary: Logout and clear refresh token
      description: Logs out the user and clears the refresh token cookie.
      responses:
        200:
          description: Successfully logged out
          content:
            application/json:
              example:
                message: "Logged out successfully"
        500:
          description: Server error
components:
  securitySchemes:
    BearerAuth:
      type: http
      scheme: bearer
      bearerFormat: JWT
security:
  - BearerAuth: []
```

5. Išvados

Šio projekto metu sukurta patogi ir saugi platforma, skirta darbo skelbimų ir komentarų valdymui, išskirianti 3 naudotojų roles: svečius, registruotus naudotojus ir administratorius. Naudojant JWT autentifikaciją bei autorizaciją, užtikrinamas aiškus prieigos valdymas, o administratoriai gali visapusiškai tvarkyti skelbimus, kategorijas ir komentarus. Sistema įgyvendina būtinas CRUD operacijas, saugo duomenis MongoDB duomenų bazėje ir turi patogią naudotojo sąsają. Projektas yra talpinamas „Google Cloud“ infrastruktūroje ir „Vercel“ platformoje, užtikrinant lankstų diegimą.