

Document: Sviluppo del Progetto

Revision: 1.0



UNIVERSITY  
OF TRENTO

Dipartimento di Ingegneria e Scienza dell'Informazione

# ShopGreen

## Sviluppo della web app ShopGreen

Doc. Name	D2-ShopGreen	Doc. Number	D2 V1.0
Description	Documento di sviluppo dell'applicazione		

## INDICE

Sviluppo della web app ShopGreen	1
INDICE	1
<b>1. Scopo del documento</b>	<b>2</b>
<b>2. Web APIs</b>	<b>2</b>
<b>3. Implementation</b>	<b>45</b>
3.1. Repository Organization	46
3.2. Branching strategy e organizzazione del lavoro	46
3.3. Dependencies	47
3.4. Database	47
3.5. Testing	47
<b>4. Front End</b>	<b>56</b>
<b>5. Deployment</b>	<b>61</b>

Document:	Descrizione di Progetto
Revision:	1.0

## 1. Scopo del documento

Il presente documento riporta tutte le informazioni necessarie per lo sviluppo delle APIs e di altre componenti dell'applicazione ShopGreen. In particolare, presenta tutti i framework, intesi come software di vario tipo, necessari per realizzare i servizi di consultazione e gestione delle attività sostenibili sul territorio di Trento con l'applicazione. Vengono affrontate le web APIs, l'organizzazione dei file e del lavoro, le dependencies, il database e il testing. Infine vengono presentati Front End e Deployment.

## 2. Web APIs

Le APIs sono state documentate secondo le specifiche OpenApi 3.0.1 e la documentazione è consultabile pubblicamente su swagger all'indirizzo [Swagger-APIs](#).

Le APIs sono state progettate seguendo i principi restful e sulla base dei modelli realizzati per il database con la gestione di diverse entità e dei loro attributi specifici. L'Access Control è stato ideato tenendo conto dei ruoli diversi degli utenti e dei loro privilegi sfruttando lo standard JWT per autenticazioni e autorizzazioni. Sono stati previsti diversi tipi di richieste e sotto richieste in modo da fornire una moltitudine di operazioni che rendesse la navigazione quanto più accomodante agli utenti. Le risposte di errore sono state inoltre standardizzate per raccogliere i casi problematici sotto gli ambiti principali che riguardano il progetto per renderne facile la comprensione.

La specifica delle APIs è disponibile al link [GitHub-APIs](#). Il contenuto del file .yaml (superset di JSON) è qui riportato per esteso:

```
JSON
openapi: 3.0.1
info:
  title: Default module
  description: ''
  version: 1.0.0
tags: []
paths:
  /auth/login:
    post:
```

```
summary: /auth/login
deprecated: false
description: >-
    Autentica un utente o operatore usando username e password e
restituisce
    JWT token
tags: []
parameters: []
requestBody:
    content:
        application/json:
            schema:
                type: object
                properties:
                    username:
                        type: string
                        description: username dell'utente
                    password:
                        type: string
                        description: password dell'utente
                required:
                    - username
                    - password
            example:
                username: anna
                password: anna
        required: true
responses:
    '200':
        description: Autenticazione riuscita
        content:
            application/json:
                schema:
                    type: object
                    properties:
                        success:
                            type: boolean
                            description: True se autenticazione è riuscita
                        titolo:
                            type: string
                        dettagli:
                            type: string
                        username:
                            type: string
                        id:
                            type: string
                        token:
                            type: string
                            description: JWT token
                        ruolo:
                            type: string
                            enum:
                                - utente
                                - operatore
                                - venditore
                description: >-
                    Ruolo per il reindirizzamento (utente,venditore o
```

```
        operatore)
    self:
      type: string
  required:
    - success
    - token
    - ruolo
    - titolo
    - dettagli
    - username
    - id
    - self
  headers: {}
'400':
  description: Username e/o password mancanti.
  content:
    application/json:
      schema:
        title: ''
        type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
            type: string
            description: titolo dell'errore
          dettagli:
            type: string
            description: messaggio di errore
        required:
          - success
          - titolo
          - dettagli
      headers: {}
'401':
  description: Autenticazione fallita. Utente non trovato.
  content:
    application/json:
      schema:
        title: ''
        type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
            type: string
            description: titolo dell'errore
          dettagli:
            type: string
            description: messaggio di errore
        required:
          - success
          - titolo
          - dettagli
      headers: {}
```

```
'500':
  description: Il server fallisce nel stabilire una connessione con il
database
  content:
    application/json:
      schema:
        title: ''
        type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
            type: string
            description: titolo dell'errore
          dettagli:
            type: string
            description: messaggio di errore
        required:
          - success
          - titolo
          - dettagli
      headers: {}
  security: []
/negozi:
  get:
    summary: /negozi
    deprecated: false
    description: >-
      Ricerca e visualizzazione di tutti i segnaposti relativi alle attività
      commerciali oppure delle segnalazioni di nuove attività /
rivendicazioni
    tags: []
    parameters:
      - name: nome
        in: query
        description: nome del negozio da cercare
        required: false
        example: Naturasi
        schema:
          type: string
      - name: categoria
        in: query
        description: categoria dei negozi da visualizzare
        required: false
        example: alimenti
        schema:
          type: string
      - name: verificatoDaOperatore
        in: query
        description: se l'attività è stata verificata da un operatore
        required: false
        example: 'true'
        schema:
          type: boolean
          default: false
      - name: proprietarioInAttesa
```

```
in: query
description: >-
    id del proprietario che ha rivendicato la sua attività (già
    esistente o meno)
required: false
example: 4547834hbhfsjh65464nbjkd
schema:
    type: string
- name: autenticazione
in: header
description: ''
required: false
example: ''
schema:
    type: string
responses:
'200':
description: >-
    La ricerca ha successo, restituisce un array con i negozi
    corrispondenti alla ricerca oppure le segnalazioni
content:
    application/json:
        schema:
            type: array
            items:
                type: object
                properties:
                    _id:
                        type: string
                    nome:
                        type: string
                    categoria:
                        type: array
                        items:
                            type: string
                    coordinate:
                        type: array
                        items:
                            type: number
        verificatoDaOperatore:
            type: boolean
        maps:
            type: string
            nullable: true
        mappe:
            type: string
            nullable: true
        linkSito:
            type: string
            nullable: true
        licenzaOppureFoto:
            type: string
        orari:
            $ref: '#/components/schemas/orari'
        sostenibilitàVerificata:
            type: boolean
        proprietario:
```

```
        type: string
    proprietarioInAttesa:
        type: string
    required:
        - _id
        - nome
        - categoria
        - coordinate
        - verificatoDaOperatore
        - licenzaOppureFoto
        - sostenibilitàVerificata
        - proprietario
        - proprietarioInAttesa
    headers: {}
'500':
    description: Il server fallisce nel stabilire una connessione con il
database
    content:
        application/json:
            schema:
                title: ''
                type: object
            properties:
                success:
                    type: boolean
                    description: codice di stato
                titolo:
                    type: string
                    description: titolo dell'errore
                dettagli:
                    type: string
                    description: messaggio di errore
            required:
                - success
                - titolo
                - dettagli
        headers: {}
        security:
            - Autenticazione: []
post:
    summary: /negozi
    deprecated: false
    description: Creazione istanza nuova attività da verificare per
l'operatore
    tags: []
    parameters:
        - name: autenticazione
          in: header
          description: ''
          required: false
          example: >-
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFubmEiLCJpZCI6IjY5MzBjMD
R1YTk1MGY0ZmJlYWVhOWJkMCIsInJ1b2xvIjoib3BlcmF0b3JlIiwiaWF0IjoxNzY2NTk3NzcyLCJle
HAIoje3NjY20DQxNzJ9.MA-wanX5gpqY0-5To20DApgUopeEZweFCan5sIpLLU
            schema:
                type: string
```

```
requestBody:  
  content:  
    application/json:  
      schema:  
        type: object  
        properties:  
          nome:  
            type: string  
          coordinate:  
            type: array  
            items:  
              type: number  
              nullable: true  
          categoria:  
            type: array  
            items:  
              type: string  
          sostenibilitàVerificata:  
            type: boolean  
            default: false  
          maps:  
            type: string  
            nullable: true  
          mappe:  
            type: string  
            nullable: true  
          linkSito:  
            type: string  
            nullable: true  
          orari:  
            allOf:  
              - $ref: '#/components/schemas/orari'  
            nullable: true  
          licenzaOppureFoto:  
            type: string  
          verificatoDaOperatore:  
            type: boolean  
            default: false  
            readOnly: true  
          proprietario:  
            type: string  
            description: true se l'utente che sta segnalando è il  
proprietario  
            required:  
              - nome  
              - categoria  
              - sostenibilitàVerificata  
              - licenzaOppureFoto  
              - verificatoDaOperatore  
              - proprietario  
            example: ''  
          required: true  
        responses:  
          '201':  
            description: >-  
              L'invio è riuscito, i dati necessari sono stati inseriti ed il db è  
              stato raggiunto correttamente
```

```
content:
  application/json:
    schema:
      type: object
      properties:
        success:
          type: boolean
          description: true se invio riuscito, false altrimenti
        required:
          - success
      example:
        success: false
    headers: {}
'400':
  description: >-
    Richiesta che non rispetta i criteri necessari, mancano campi
    necessari
  content:
    application/json:
      schema:
        title: ''
        type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
            type: string
            description: titolo dell'errore
          dettagli:
            type: string
            description: messaggio di errore
        required:
          - success
          - titolo
          - dettagli
    headers: {}
'401':
  description: >-
    Il token non è valido, è scaduto o non è possibile controllarne
    l'autenticità
  content:
    application/json:
      schema:
        title: ''
        type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
            type: string
            description: titolo dell'errore
          dettagli:
            type: string
            description: messaggio di errore
        required:
```

```
        - success
        - titolo
        - dettagli
    example:
        success: false
        titolo: Mancata autenticazione
        dettagli: Per segnalare una nuova attività esegui l'accesso o
registrati
    headers: {}
'403':
    description: Il token non fornisce i permessi richiesti
    content:
        application/json:
            schema:
                title: ''
                type: object
            properties:
                success:
                    type: boolean
                    description: codice di stato
                titolo:
                    type: string
                    description: titolo dell'errore
                dettagli:
                    type: string
                    description: messaggio di errore
            required:
                - success
                - titolo
                - dettagli
    headers: {}
'500':
    description: Il server fallisce nel stabilire una connessione con il
database
    content:
        application/json:
            schema:
                title: ''
                type: object
            properties:
                success:
                    type: boolean
                    description: codice di stato
                titolo:
                    type: string
                    description: titolo dell'errore
                dettagli:
                    type: string
                    description: messaggio di errore
            required:
                - success
                - titolo
                - dettagli
    headers: {}
    security:
        - Autenticazione: []
/negozi/{negozi_id}:
```

```
get:
  summary: /negozi/{negozi_id}
  deprecated: false
  description: >-
    Visualizzazione delle informazioni relative ad un negozio/segnalazione
    mediante il suo id
  tags: []
  parameters:
    - name: negozio_id
      in: path
      description: ''
      required: true
      example: 694ace8252a0cde59e87d06e
      schema:
        type: string
    - name: autenticazione
      in: header
      description: ''
      required: false
      example: >-
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmcFtZSI6ImFubmEiLCJpZCI6IjY5MzBjMD
R1YTk1MGY0ZmJ1YWVh0WJkMCIsInJ1b2xvIjoib3BlcmF0b3JlIiwiaWF0IjoxNzY2NTExNjYxLCJle
HAI0jE3NjY10TgwNjF9.9Inx0aEn7dr9u6xxAd9k_1xDY0x6SuJvCHmJd0aDRk0
      schema:
        type: string
  responses:
    '200':
      description: >-
        La richiesta ha successo, restituisce le informazioni relative al
        negozio/segnalazione selezionato/a
      content:
        application/json:
          schema:
            title: ''
            type: object
            properties:
              negozio_id:
                type: string
              nome:
                type: string
              categoria:
                type: array
                items:
                  type: string
                  nullable: true
              sostenibilitàVerificata:
                type: boolean
                default: false
              maps:
                type: string
                nullable: true
              mappe:
                type: string
                nullable: true
              linkSito:
                type: string
```

```
        nullable: true
licenzaOppureFoto:
  type: string
verificatoDaOperatore:
  type: string
coordinate:
  type: array
  items:
    type: number
    minItems: 2
    maxItems: 2
proprietario:
  type: string
proprietarioInAttesa:
  type: string
orari:
  $ref: '#/components/schemas/orari'
required:
- negozio_id
- nome
- sostenibilitàVerificata
- verificatoDaOperatore
- licenzaOppureFoto
- coordinate
- proprietario
- proprietarioInAttesa
- orari
headers: {}
'404':
  description: Negozio non trovato
  content:
    application/json:
      schema:
        title: ''
        type: object
      properties:
        success:
          type: boolean
          description: codice di stato
        titolo:
          type: string
          description: titolo dell'errore
        dettagli:
          type: string
          description: messaggio di errore
      required:
        - success
        - titolo
        - dettagli
  headers: {}
'500':
  description: Il server fallisce nel stabilire una connessione con il
database
  content:
    application/json:
      schema:
        title: ''
```

```
type: object
properties:
  success:
    type: boolean
    description: codice di stato
  titolo:
    type: string
    description: titolo dell'errore
  dettagli:
    type: string
    description: messaggio di errore
required:
- success
- titolo
- dettagli
headers: {}
security:
- Autenticazione: []
delete:
  summary: /negozi/{negozi_id}
  deprecated: false
  description: >-
    Elimina un negozio (licenza non presente o non valida, segnalazioni, ecc.), eseguibile solo da un operatore
  tags: []
  parameters:
    - name: negozi_id
      in: path
      description: ''
      required: true
      example: 694c1ed06bbadd7c8d451993
      schema:
        type: string
    - name: autenticazione
      in: header
      description: ''
      required: false
      example: >-
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFubmEiLCJpZCI6IjY5MzBjMD
R1YTk1MGY0ZmJlYWVhOWJkMCIsInJ1b2xvIjoib3BlcmF0b3JlIiwiaWF0IjoxNzY2NTExNjYxLCJle
HAI0jE3NjY1OTgwNjF9.9Inx0aEn7dr9u6xxAd9k_1xDY0x6SuJvCHmJd0aDRk0
      schema:
        type: string
  responses:
    '200':
      description: L'operazione ha successo e viene inviato un messaggio di conferma
      content:
        application/json:
          schema:
            title: ''
            type: object
            properties:
              Success:
                type: boolean
            required:
```

```
        - Success
    example:
      Success: true
    headers: {}
'204':
  description: La richiesta ha successo
  headers: {}
'401':
  description: >-
    Il token non è valido, è scaduto o non è possibile controllarne
    l'autenticità
  content:
    application/json:
      schema:
        title: ''
        type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
            type: string
            description: titolo dell'errore
          dettagli:
            type: string
            description: messaggio di errore
        required:
          - success
          - titolo
          - dettagli
  example:
    success: false
    titolo: Mancata autenticazione
    dettagli: >-
      Questa operazione è ristretta ai soli operatori o proprietari
      del negozio, fai l'accesso per procedere
  headers: {}
'403':
  description: Il token non fornisce i permessi richiesti
  content:
    application/json:
      schema:
        title: ''
        type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
            type: string
            description: titolo dell'errore
          dettagli:
            type: string
            description: messaggio di errore
        required:
          - success
          - titolo
```

```
        - dettagli
example:
  success: false
  titolo: Unauthorized
  dettagli: >-
    Questo account non ha i permessi per procedere con
    l'operazione
headers: {}
'404':
  description: Il negozio da eliminare non è stato trovato
content:
  application/json:
    schema:
      title: ''
      type: object
    properties:
      success:
        type: boolean
        description: codice di stato
      titolo:
        type: string
        description: titolo dell'errore
      dettagli:
        type: string
        description: messaggio di errore
    required:
      - success
      - titolo
      - dettagli
headers: {}
'500':
  description: Il server fallisce nel stabilire una connessione con il
database
content:
  application/json:
    schema:
      title: ''
      type: object
    properties:
      success:
        type: boolean
        description: codice di stato
      titolo:
        type: string
        description: titolo dell'errore
      dettagli:
        type: string
        description: messaggio di errore
    required:
      - success
      - titolo
      - dettagli
example:
  success: true
  titolo: Server non raggiungibile
  dettagli: >-
    Al momento il server non è raggiungibile, per cui non è
```

```
        possibile completare l'operazione, per favore ritenta
    headers: {}
  security:
    - Autenticazione: []
put:
  summary: /negozi/{negozi_id}
  deprecated: false
  description: >-
    Modifica i campi di una segnalazione di una nuova attività o di una
    rivendicazione sovrascrivendo quelli preesistenti
  tags: []
  parameters:
    - name: negozio_id
      in: path
      description: ''
      required: true
      example: 694c25228eed813a9305ae70
      schema:
        type: string
    - name: autenticazione
      in: header
      description: ''
      required: false
      example: >-
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFubmEiLCJpZCI6IjY5MzBjMD
R1YTk1MGY0ZmJ1YWVhOWJkMCIsInJ1b2xvIjoib3BlcmF0b3JlIiwiaWF0IjoxNzY2NjAwOTcxLCJle
HAI0jE3NjY20DczNzF9.FmPZ03gmdRd50sEW_N9H3ybSJvK_dfunqbrlQkQt_1k
      schema:
        type: string
  requestBody:
    content:
      application/json:
        schema:
          type: object
          properties:
            nome:
              type: string
            coordinate:
              type: array
              items:
                type: number
            categoria:
              type: array
              items:
                type: string
            sostenibilitàVerificata:
              type: boolean
              default: false
            maps:
              type: string
              nullable: true
            mappe:
              type: string
              nullable: true
            linkSito:
              type: string
```

```
        nullable: true
    orari:
      $ref: '#/components/schemas/orari'
    licenzaOppureFoto:
      type: string
    verificatoDaOperatore:
      type: boolean
    proprietario:
      type: string
      description: id del proprietario
      nullable: true
    proprietarioInAttesa:
      type: string
    required:
      - nome
      - coordinate
      - categoria
      - sostenibilitàVerificata
      - orari
      - licenzaOppureFoto
      - verificatoDaOperatore
      - proprietarioInAttesa
    examples: {}
    required: true
  responses:
    '200':
      description: La modifica ha successo, restituisce il negozio così
modificato
      content:
        application/json:
          schema:
            type: object
            properties:
              nome:
                type: string
              coordinate:
                type: array
                items:
                  type: number
              categoria:
                type: array
                items:
                  type: string
              sostenibilitàVerificata:
                type: boolean
                default: false
              maps:
                type: string
                nullable: true
              mappe:
                type: string
                nullable: true
              linkSito:
                type: string
                nullable: true
              orari:
                $ref: '#/components/schemas/orari'
```

```
    licenzaOppureFoto:
      type: string
    verificatoDaOperatore:
      type: boolean
      default: true
    proprietario:
      type: string
      description: id del proprietario
    proprietarioInAttesa:
      type: string
  required:
    - nome
    - coordinate
    - categoria
    - sostenibilitàVerificata
    - orari
    - licenzaOppureFoto
    - verificatoDaOperatore
    - proprietarioInAttesa
    - proprietario
  headers: {}
'400':
  description: L'operatore ha inviato campi non validi o mancanti
  content:
    application/json:
      schema:
        title: ''
        type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
            type: string
            description: titolo dell'errore
          dettagli:
            type: string
            description: messaggio di errore
      required:
        - success
        - titolo
        - dettagli
  headers: {}
'401':
  description: >-
    Il token non è valido, è scaduto o non è possibile controllarne
    l'autenticità
  content:
    application/json:
      schema:
        title: ''
        type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
```

```
        type: string
        description: titolo dell'errore
    dettagli:
        type: string
        description: messaggio di errore
    required:
        - success
        - titolo
        - dettagli
    example:
        success: false
        titolo: Mancata autenticazione
        dettagli: >-
            Questa operazione è ristretta ai soli operatori, fai
l'accesso
            per procedere
    headers: {}
'403':
    description: Il token non fornisce i permessi richiesti
    content:
        application/json:
            schema:
                title: ''
                type: object
                properties:
                    success:
                        type: boolean
                        description: codice di stato
                    titolo:
                        type: string
                        description: titolo dell'errore
                    dettagli:
                        type: string
                        description: messaggio di errore
            required:
                - success
                - titolo
                - dettagli
    headers: {}
'404':
    description: Negozio non trovato
    content:
        application/json:
            schema:
                title: ''
                type: object
                properties:
                    success:
                        type: boolean
                        description: codice di stato
                    titolo:
                        type: string
                        description: titolo dell'errore
                    dettagli:
                        type: string
                        description: messaggio di errore
            required:
```

```
        - success
        - titolo
        - dettagli
    headers: {}
'500':
  description: Il server fallisce nel stabilire una connessione con il
database
  content:
    application/json:
      schema:
        title: ''
        type: object
      properties:
        success:
          type: boolean
          description: codice di stato
        titolo:
          type: string
          description: titolo dell'errore
        dettagli:
          type: string
          description: messaggio di errore
      required:
        - success
        - titolo
        - dettagli
    example:
      success: true
      titolo: Server non raggiungibile
      dettagli: >-
        Al momento il server non è raggiungibile, per cui non è
        possibile completare l'operazione, per favore ritenta
    headers: {}
  security:
    - Autenticazione: []
/preferiti/users/{user_id}:
  post:
    summary: /preferiti/users/{user_id}
    deprecated: false
    description: >-
      Modifica di un campo di user, in questo caso creato per aggiungere le
      attività commerciali fisiche dalla lista dei preferiti
    tags: []
    parameters:
      - name: user_id
        in: path
        description: ''
        required: true
      example: 6930c04ea950f4fbeaea9bd0
      schema:
        type: string
      - name: autenticazione
        in: header
        description: ''
        required: false
      example: >-
```

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFubmEiLCJpZCI6IjY5MzBjMD
R1YTk1MGY0ZmJ1YWVhOWJkMCIsInJ1b2xvIjoib3BlcmF0b3JlIiwiaWF0IjoxNzY2NTExODI2LCJle
HAiOjE3NjY2MDAyMjZ9.TuGozWNE8KGV-LO_4qRF0KiQ0aU9Ss5ynz0yCjykmLA
  schema:
    type: string
  requestBody:
    content:
      application/json:
        schema:
          type: object
          properties:
            negozio_id:
              type: string
            required:
              - negozio_id
        example:
          negozio_id: 694ace8252a0cde59e87d06e
        required: true
    responses:
      '200':
        description: ''
        content:
          application/json:
            schema:
              type: object
              properties:
                success:
                  type: boolean
                  description: >-
                    Conferma la risucita dell'operazione, il negozio potrebbe
                    già essere presente nella lista
            required:
              - success
        example:
          success: true
        headers: {}
      '201':
        description: ''
        content:
          application/json:
            schema:
              title: ''
              type: object
              properties:
                success:
                  type: boolean
                  description: True se aggiunto per la prima volta
            required:
              - success
        example:
          success: true
        headers: {}
      '401':
        description: >-
          Il token non è valido, è scaduto o non è possibile controllarne
          l'autenticità
```

```
content:
  application/json:
    schema:
      title: ''
      type: object
    properties:
      success:
        type: boolean
        description: codice di stato
      titolo:
        type: string
        description: titolo dell'errore
      dettagli:
        type: string
        description: messaggio di errore
    required:
      - success
      - titolo
      - dettagli
  example:
    success: false
    titolo: Mancata autenticazione
    dettagli: >-
      Per aggiungere un'attività ai preferiti esegui l'accesso o
      registrati
  headers: {}
'403':
  description: Il token non fornisce i permessi richiesti
  content:
    application/json:
      schema:
        title: ''
        type: object
      properties:
        success:
          type: boolean
          description: codice di stato
        titolo:
          type: string
          description: titolo dell'errore
        dettagli:
          type: string
          description: messaggio di errore
      required:
        - success
        - titolo
        - dettagli
  headers: {}
'404':
  description: Utente non trovato
  content:
    application/json:
      schema:
        title: ''
        type: object
      properties:
        success:
```

```
        type: boolean
        description: codice di stato
    titolo:
        type: string
        description: titolo dell'errore
    dettagli:
        type: string
        description: messaggio di errore
    required:
        - success
        - titolo
        - dettagli
    headers: {}
database
'500':
    description: Il server fallisce nel stabilire una connessione con il
content:
    application/json:
        schema:
            title: ''
            type: object
        properties:
            success:
                type: boolean
                description: codice di stato
            titolo:
                type: string
                description: titolo dell'errore
            dettagli:
                type: string
                description: messaggio di errore
        required:
            - success
            - titolo
            - dettagli
example:
    success: false
    titolo: Server non disponibile
    dettagli: >-
        Il negozio non può essere aggiunto ai preferiti poiché il
        server non è raggiungibile
    headers: {}
security:
    - Autenticazione: []
get:
    summary: /preferiti/users/{user_id}
    deprecated: false
    description: >-
        Nell'area dedicata l'utente può vedere la lista dei suoi negozi fisici
        preferiti
    tags: []
    parameters:
        - name: user_id
          in: path
          description: ''
          required: true
    example: 6930c04ea950f4fbeaea9bd0
```

```
schema:
  type: string
- name: autenticazione
in: header
description: ''
required: false
example: >-
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmcFtZSI6ImFubmEiLCJpZCI6IjY5MzBjMD
R1YTk1MGY0ZmJ1YWVhOWJkMCIsInJ1b2xvIjoib3BlcmF0b3JlIiwiaWF0IjoxNzY2NTExODI2LCJle
HAI0jE3NjY2MDAyMjZ9.TuGozWNE8KGV-L0_4qRF0KiQ0aU9Ss5ynz0yCjykmLA
  schema:
    type: string
  responses:
    '200':
      description: >-
        La ricerca ha successo, restituisce un array con i negozi segnati
        come preferiti dall'utente
      content:
        application/json:
          schema:
            type: array
            items:
              type: object
              properties:
                negozio_id:
                  type: string
                  description: L'id per collegarsi sulla mappa
                nome:
                  type: string
                  description: Il nome per la visualizzazione
                coordinate:
                  type: string
                sostenibilitàVerificata:
                  type: string
                orari:
                  $ref: '#/components/schemas/orari'
      required:
        - negozio_id
        - nome
        - orari
        - coordinate
        - sostenibilitàVerificata
    headers: {}
    '401':
      description: Token mancante. Utente non autenticato.
      content:
        application/json:
          schema:
            title: ''
            type: object
            properties:
              success:
                type: boolean
                description: codice di stato
              titolo:
                type: string
```

```
        description: titolo dell'errore
        dettagli:
          type: string
          description: messaggio di errore
        required:
          - success
          - titolo
          - dettagli
      example:
        success: false
        titolo: Mancata autenticazione
        dettagli: >-
          Per aggiungere un'attività ai preferiti esegui l'accesso o
          registrati
      headers: {}
  '403':
    description: Il token non fornisce i permessi richiesti
    content:
      application/json:
        schema:
          title: ''
          type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
            type: string
            description: titolo dell'errore
          dettagli:
            type: string
            description: messaggio di errore
        required:
          - success
          - titolo
          - dettagli
      headers: {}
  '404':
    description: Utente non trovato
    content:
      application/json:
        schema:
          type: object
        properties: {}
    headers: {}
  '500':
    description: Il server fallisce nel stabilire una connessione con il
database
    content:
      application/json:
        schema:
          title: ''
          type: object
        properties:
          success:
            type: boolean
            description: codice di stato
```

```
titolo:
    type: string
    description: titolo dell'errore
dettagli:
    type: string
    description: messaggio di errore
required:
- success
- titolo
- dettagli
example:
    success: false
    titolo: Server non raggiungibile
    dettagli: >-
        Al momento il server non è raggiungibile, per cui non è
        possibile completare l'operazione, per favore ritenta
    headers: {}
security:
- Autenticazione: []
delete:
    summary: /preferiti/users/{user_id}
    deprecated: false
    description: Serve a rimuovere un negozio fisico dalla lista dei
preferiti
    tags: []
    parameters:
- name: user_id
    in: path
    description: ''
    required: true
    example: 6930c04ea950f4fbeaea9bd0
    schema:
        type: string
- name: negozio_id
    in: query
    description: ''
    required: false
    example: 694ad3c20b1b8c4c33a6ca49
    schema:
        type: string
- name: autenticazione
    in: header
    description: ''
    required: false
    example: >-
schema:
    type: string
responses:
'204':
    description: >-
        L'operazione è andata a buon fine ma non si vuole mandare alcun
        messaggio
    headers: {}
```

```
'400':
  description: Manca l'id del negozio da rimuovere dai preferiti
  content:
    application/json:
      schema:
        type: object
        properties: {}
  headers: {}

'401':
  description: >-
    Il token non è valido, è scaduto o non è possibile controllarne
    l'autenticità
  content:
    application/json:
      schema:
        title: ''
        type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
            type: string
            description: titolo dell'errore
          dettagli:
            type: string
            description: messaggio di errore
        required:
          - success
          - titolo
          - dettagli
  headers: {}

'403':
  description: Il token non fornisce i permessi richiesti
  content:
    application/json:
      schema:
        title: ''
        type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
            type: string
            description: titolo dell'errore
          dettagli:
            type: string
            description: messaggio di errore
        required:
          - success
          - titolo
          - dettagli
  headers: {}

'404':
  description: >-
```

rimosso

Se il negozio che si vuole rimuovere dalla lista è già stato  
ma per qualche motivo l'interfaccia non è stata aggiornata, va  
segnalato che non è possibile rimuoverlo nuovamente poiché nel  
database non è più associato all'utente

```
content:
  application/json:
    schema:
      title: ''
      type: object
    properties:
      success:
        type: boolean
        description: codice di stato
      titolo:
        type: string
        description: titolo dell'errore
      dettagli:
        type: string
        description: messaggio di errore
    required:
      - success
      - titolo
      - dettagli
  example:
    success: false
    titolo: Not found
    dettagli: >-
      il negozio che si tenta di rimuovere dalla lista è già stato
      rimosso o non è presente nella lista
  headers: {}
```

'500':

description: Il server fallisce nel stabilire una connessione con il  
database

```
content:
  application/json:
    schema:
      title: ''
      type: object
    properties:
      success:
        type: boolean
        description: codice di stato
      titolo:
        type: string
        description: titolo dell'errore
      dettagli:
        type: string
        description: messaggio di errore
    required:
      - success
      - titolo
      - dettagli
  example:
    success: false
    titolo: Server non raggiungibile
    dettagli: >-
```

```
Al momento il server non è raggiungibile, per cui non è
possibile completare l'operazione, per favore ritenta
headers: {}
security:
- Autenticazione: []
/feedback:
post:
summary: /feedback
deprecated: false
description: >-
    permette ad un utente di inviare un feedback sulla sostenibilità di un
    negozio verificato da un operatore ma col segnaposto ancora grigio
tags: []
parameters:
- name: autenticazione
  in: header
  description: ''
  required: false
example: >-
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmcFtZSI6ImFubmEiLCJpZCI6IjY5MzBjMD
R1YTk1MGY0ZmJ1YWVh0WJkMCIsInJ1b2xvIjoib3BlcmF0b3JlIiwiaWF0IjoxNzY2NTk10DE1LCJle
HAi0jE3NjY20DIyMTV9.YfzjKUM-UCWhSJnbH1IXe_5C5qECCEY7_uMbC4zW9wY
schema:
  type: string
requestBody:
content:
application/json:
schema:
  type: object
  properties:
    negozio:
      type: string
    feedback:
      type: boolean
  required:
  - negozio
  - feedback
example:
  negozio: 694c249f8eed813a9305ae50
  feedback: 'true'
required: true
responses:
'200':
description: >-
    Il feedback è stato registrato e viene inviato un messaggio di
    conferma
content:
application/json:
schema:
  type: object
  properties:
    success:
      type: boolean
      description: True se ha avuto successo
  required:
  - success
```

```
example:
  success: true
headers: {}
'201':
  description: >-
    Il feedback è stato registrato e viene inviato un messaggio di conferma, inoltre viene restituito l'id del negozio col numero di feedback raggiunti sia positivi che negativi
content:
  application/json:
    schema:
      title: ''
      type: object
    properties:
      success:
        type: boolean
        description: True se ha avuto successo
      positive:
        type: integer
        description: numero di feedback positivi, compreso quello registrato
        minimum: 0
        maximum: 8
      negative:
        type: integer
        description: numero di feedback negativi, compreso quello registrato
        minimum: 0
        maximum: 8
      required:
        - success
        - positive
        - negative
    example:
      success: false
      positive: 4
      negative: 8
    headers: {}
'401':
  description: Utente non trovato
content:
  application/json:
    schema:
      title: ''
      type: object
    properties:
      success:
        type: boolean
        description: codice di stato
      titolo:
        type: string
        description: titolo dell'errore
      dettagli:
        type: string
        description: messaggio di errore
    required:
      - success
```

```
        - titolo
        - dettagli
    headers: {}
'403':
  description: Il token non fornisce i permessi richiesti
  content:
    application/json:
      schema:
        title: ''
        type: object
      properties:
        success:
          type: boolean
          description: codice di stato
        titolo:
          type: string
          description: titolo dell'errore
        dettagli:
          type: string
          description: messaggio di errore
      required:
        - success
        - titolo
        - dettagli
    headers: {}
'409':
  description: >-
    L'utente ha già mandato un feedback per quel negozio, ma per errore
    di sincronizzazione dell'interfaccia prova a mandarne un altro
  content:
    application/json:
      schema:
        title: ''
        type: object
      properties:
        success:
          type: boolean
          description: codice di stato
        titolo:
          type: string
          description: titolo dell'errore
        dettagli:
          type: string
          description: messaggio di errore
      required:
        - success
        - titolo
        - dettagli
    example:
      success: false
      titolo: Feedback già inviato
      dettagli: Non è possibile inviare più di un feedback per
attività
      headers: {}
'500':
  description: Il server fallisce nel stabilire una connessione con il
database
```

```
content:
  application/json:
    schema:
      title: ''
      type: object
      properties:
        success:
          type: boolean
          description: codice di stato
        titolo:
          type: string
          description: titolo dell'errore
        dettagli:
          type: string
          description: messaggio di errore
      required:
        - success
        - titolo
        - dettagli
    example:
      success: false
      titolo: Server non raggiungibile
      dettagli: >-
        Al momento il server non è raggiungibile, per cui non è
        possibile completare l'operazione, per favore ritenta
    headers: {}
    security:
      - Autenticazione: []
/feedback/:
get:
  summary: /feedback
  deprecated: false
  description: >-
    Visualizzazione del feedback inviato da un utente ad un determinato
    negozio
  tags: []
  parameters:
    - name: negozio_id
      in: query
      description: ''
      required: false
      example: 694c249f8eed813a9305ae50
      schema:
        type: string
    - name: autenticazione
      in: header
      description: ''
      required: false
      example: >-
        schema:
          type: string
  responses:
    '200':
```

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFubmEiLCJpZCI6IjY5MzBjMDR1YTk1MGY0ZmJlYWVhOWJkMCIsInJ1b2xvIjoib3BlcmF0b3JlIiwiaWF0IjoxNzY2NTk1ODE1LCJleHAiOjE3NjY20DIyMTV9.YfZjKUM-UCWhSJnbH1IXe\_5C5qECCEY7\_uMbC4zW9wY

schema:

type: string

responses:

'200':

```
description: Il feedback esiste e viene recuperato con successo
content:
  application/json:
    schema:
      type: object
    properties:
      success:
        type: boolean
        description: True come conferma che l'operazione è riuscita
      feedback:
        type: boolean
        description: >-
          True se la votazione lasciata era positiva, false se era
          negativa
    required:
      - feedback
      - success
    example:
      success: true
      feedback: true
  headers: {}
'400':
  description: Manca l'id del negozio
  content:
    application/json:
      schema:
        title: ''
        type: object
      properties:
        success:
          type: boolean
          description: codice di stato
        titolo:
          type: string
          description: titolo dell'errore
        dettagli:
          type: string
          description: messaggio di errore
    required:
      - success
      - titolo
      - dettagli
  headers: {}
'401':
  description: Utente non trovato
  content:
    application/json:
      schema:
        title: ''
        type: object
      properties:
        success:
          type: boolean
          description: codice di stato
        titolo:
          type: string
          description: titolo dell'errore
```

```
        dettagli:
          type: string
          description: messaggio di errore
        required:
          - success
          - titolo
          - dettagli
      headers: {}
'403':
  description: Il token non fornisce i permessi richiesti
  content:
    application/json:
      schema:
        title: ''
        type: object
      properties:
        success:
          type: boolean
          description: codice di stato
        titolo:
          type: string
          description: titolo dell'errore
        dettagli:
          type: string
          description: messaggio di errore
      required:
        - success
        - titolo
        - dettagli
    headers: {}
'404':
  description: Non è presente alcun feedback associato a questi user e
negozio
  content:
    application/json:
      schema:
        title: ''
        type: object
      properties:
        success:
          type: boolean
          description: >-
            True se la ricerca è stata completata ma il feedback non
è
            stato trovato, false se non è stato possibile concludere
            la ricerca e dunque il feedback potrebbe esserci ma non
            essere stato trovato
      required:
        - success
      example:
        success: true
    headers: {}
'500':
  description: Il server fallisce nel stabilire una connessione con il
database
  content:
    application/json:
```

```
schema:
  title: ''
  type: object
  properties:
    success:
      type: boolean
      description: codice di stato
    titolo:
      type: string
      description: titolo dell'errore
    dettagli:
      type: string
      description: messaggio di errore
  required:
    - success
    - titolo
    - dettagli
  example:
    success: false
    titolo: Server non raggiungibile
    dettagli: >-
      Al momento il server non è raggiungibile, per cui non è
      possibile completare l'operazione, per favore ritenta
  headers: {}
  security:
    - Autenticazione: []
/ecommerce:
get:
  summary: /ecommerce
  deprecated: false
  description: >-
    Restituisce la lista dei venditori disposti ad incontrarsi nella zona
    selezionata
  tags: []
  parameters:
    - name: zona
      in: query
      description: 'filtro per zona'
      required: false
      example: Gardolo
    schema:
      type: string
    - name: categoria
      in: query
      description: filtro per categoria
      required: false
      example: vestiario
    schema:
      type: string
  responses:
    '200':
      description: La ricerca ha successo, restituisce una lista di
venditori
      content:
        application/json:
          schema:
            type: array
```

```
items:
  type: object
  properties:
    username:
      type: string
    user_id:
      type: string
    profile_picture:
      type: string
    nullable: true
  Zone:
    type: array
    items:
      type: string
    enum:
      - Meano
      - Gardolo
      - Argentario
      - Centro Storico Piedicastello
      - Bondone
      - San Giuseppe Santa Chiara
      - Sardagna
      - Povo
      - Oltrefersina
      - Ravina-Romagnano
      - Villazzano
      - Mattarello
  Links:
    type: array
    items:
      type: string
  Categorie:
    type: array
    items:
      type: string
  Info:
    type: string
  required:
    - username
    - Zone
    - Links
    - user_id
    - Categorie
  headers: {}
500':
  description: Il server fallisce nel stabilire una connessione con il
database
  content:
    application/json:
      schema:
        type: object
        properties:
          success:
            type: boolean
            description: codice di stato
          titolo:
            type: string
```

```
        description: titolo dell'errore
        dettagli:
          type: string
          description: messaggio di errore
        required:
          - success
          - titolo
          - dettagli
      headers: {}
    security: []
/ecommerce/{ecommerce_id}:
  get:
    summary: /ecommerce/{ecommerce_id}
    deprecated: false
    description: 'Visualizzazione dei dettagli di un venditore e-commerce '
    tags: []
    parameters:
      - name: ecommerce_id
        in: path
        description: ''
        required: true
        schema:
          type: string
    responses:
      '200':
        description: >-
          La richiesta ha successo, restituisce le informazioni relative al
          venditore selezionato
        content:
          application/json:
            schema:
              type: object
              properties:
                username:
                  type: string
                user_id:
                  type: string
                profile_picture:
                  type: string
            Links:
              type: array
              items:
                type: string
            Categorie:
              type: array
              items:
                type: string
            Zone:
              type: array
              items:
                type: string
            Info:
              type: string
    required:
      - username
      - Links
      - Categorie
```

```
        - Zone
        - user_id
    headers: {}
'400':
  description: L'ID fornito non è valido.
  content:
    application/json:
      schema:
        title: ''
        type: object
      properties:
        success:
          type: boolean
          description: codice di stato
        titolo:
          type: string
          description: titolo dell'errore
        dettagli:
          type: string
          description: messaggio di errore
      required:
        - success
        - titolo
        - dettagli
    example:
      success: false
      titolo: Bad Request
      dettagli: L'ID fornito non è valido
'404':
  description: Non è presente nessun ecommerce con questo ID
  content:
    application/json:
      schema:
        title: ''
        type: object
      properties:
        success:
          type: boolean
          description: codice di stato
        titolo:
          type: string
          description: titolo dell'errore
        dettagli:
          type: string
          description: messaggio di errore
      required:
        - success
        - titolo
        - dettagli
    example:
      success: true
      titolo: Not Found
      dettagli: Venditore ecommerce non trovato
'500':
  description: Il server fallisce nel stabilire una connessione con il
database
  content:
```

```
application/json:
  schema:
    title: ''
    type: object
    properties:
      success:
        type: boolean
        description: codice di stato
      titolo:
        type: string
        description: titolo dell'errore
      dettagli:
        type: string
        description: messaggio di errore
    required:
      - success
      - titolo
      - dettagli
    headers: {}
  security:
    - Autenticazione: []
components:
  schemas:
    orari:
      type: object
      properties:
        lunedì:
          type: object
          properties:
            chiuso:
              type: boolean
            slot:
              type: array
              items:
                type: object
                properties:
                  apertura:
                    type: string
                  chiusura:
                    type: string
        required:
          - chiuso
          - slot
        martedì:
          type: object
          properties:
            chiuso:
              type: boolean
            slot:
              type: array
              items:
                type: object
                properties:
                  apertura:
                    type: string
                  chiusura:
                    type: string
```

```
        required:
          - apertura
          - chiusura
    required:
      - chiuso
      - slot
mercoledi:
  type: object
  properties:
    chiuso:
      type: boolean
    slot:
      type: array
      items:
        type: object
        properties:
          apertura:
            type: string
          chiusura:
            type: string
    required:
      - apertura
      - chiusura
  required:
    - chiuso
    - slot
giovedi:
  type: object
  properties:
    chiuso:
      type: boolean
    slot:
      type: array
      items:
        type: object
        properties:
          apertura:
            type: string
          chiusura:
            type: string
    required:
      - apertura
      - chiusura
  required:
    - chiuso
    - slot
venerdi:
  type: object
  properties:
    chiuso:
      type: boolean
    slot:
      type: array
      items:
        type: object
        properties:
          apertura:
```

```
        type: string
    chiusura:
        type: string
    required:
        - apertura
        - chiusura
    required:
        - chiuso
        - slot
    sabato:
        type: object
        properties:
            chiuso:
                type: boolean
            slot:
                type: array
                items:
                    type: object
                    properties:
                        apertura:
                            type: string
                        chiusura:
                            type: string
                    required:
                        - apertura
                        - chiusura
                required:
                    - chiuso
                    - slot
    domenica:
        type: object
        properties:
            chiuso:
                type: boolean
            slot:
                type: array
                items:
                    type: object
                    properties:
                        apertura:
                            type: string
                        chiusura:
                            type: string
                    required:
                        - chiuso
                        - slot
    required:
        - lunedi
        - martedi
        - mercoledi
        - giovedi
        - venerdi
        - sabato
        - domenica
    nullable: true
responses: {}
securitySchemes:
```

```
Autenticazione:  
  type: http  
  scheme: bearer  
servers:  
  # Added by API Auto Mocking Plugin  
  - description: SwaggerHub API Auto Mocking  
    url: https://virtserver.swaggerhub.com/aaa-4ad/aaa/1.0.0  
security: []
```

### 3. Implementation

L'applicazione è stata sviluppata utilizzando le seguenti tecnologie: NodeJS, MongoDB, React. Le scelte tecnologiche sono state motivate dal materiale e dalle indicazioni fornite dal corso oltre alla consultazione con colleghi più esperti in materia.

#### 3.1. Repository Organization

Il codice del progetto è diviso in due repository, una dedicata al Back End ([GitHub-Back End](#)) e una dedicata al Front End ([GitHub-Front End](#)) entrambe strutturate e organizzate in cartelle.

##### Back End:

- .jest per importare variabili d'ambiente
- coverage collegamento ai path delle risorse
- /lcov-report gestione di alcune componenti e della grafica
- /GreenShop codice degli header delle pagine web
- /src/db codice delle diverse pagine web
- models codice degli oggetti nelle pagine web
- /public/images immagini utili
- src/db implementazione API
- /models modelli dati mongoose

##### Front End:

- components codice delle componenti per il Front End
- pages codice delle pagine per il Front End
- services collegamento Back-Front End

#### 3.2. Branching strategy e organizzazione del lavoro

Lo sviluppo è frutto dell'impegno di tutti i membri del gruppo. Il lavoro non è stato però diviso equamente in questo ambito ma lo è stato a livello complessivo del progetto, infatti le api sono state sviluppate per la maggior parte da Sofia Cestari e Anna Luvisotto, mentre il Back End nuovamente da Anna Luvisotto e da Laura Prandina che ha anche contribuito al Front End. Sono stati eseguiti quasi 90 commit anche se l'intento era di sviluppare quanto più possibile e fare commit complessivi (motivo per cui è stato comodo separare Back End e Front End per evitare spiacevoli sovrapposizioni). I commit sono così divisi tra i membri: 20 di Laura Prandina, 10 di

Document:

Descrizione di Progetto

Revision:

1.0

---

Sofia Cestari, 34 di Anna Luvisotto per il Back End; 9 di Laura Prandina, 19 di Anna Luvisotto. Alcuni commit sono stati effettuati per testare l'utilizzo dei branches di github mentre altri commit sono stati effettuati in numero elevato perché si trattava di piccole correzioni a singoli file invece che grandi caricamenti di nuovi file.

Abbiamo scelto di affidarci all'uso di un singolo branch, il main branch, per entrambe le repository poiché ci siamo rese conto che la gestione migliora evitando merge frequenti poiché anche piccole variazioni tra le versioni richiedono lunghe e attente revisioni. Dunque i branch sono stati usati collettivamente per la gestione di tutte le componenti del progetto.

Inizialmente è stato creato anche un branch di presentazione del team all'interno della repository del Back End ma è stato poi abbandonato ed eliminato. Per questo motivo nel branch sono presenti due file ma questi ultimi sono stati inseriti nella loro versione aggiornata all'interno del branch main.

### 3.3. Dependencies

Il progetto npm si basa sui seguenti moduli esterni:

- Google-auth-library – Supporto per il login con Google
- Jsonwebtoken – Autenticazione JWT
- Cors – Supporto alle chiamate Cors
- Mongoose – Libreria per interfacciarsi con MongoDB
- Express – Framework web utilizzato nel backend
- crypto – Crittografia
- nodemailer – Per servizio mail

E le seguenti dipendenze di sviluppo:

- Dotenv – Gestione variabili d'ambiente
- Jest – Testing

### 3.4. Database

Per la gestione dei dati dell'applicazione abbiamo definito sette modelli utilizzando mongoose:

1. **User:** il quale rappresenta i 3 tipi di user della nostra applicazione (ovvero: utente, venditore, operatore) e raccoglie i dati relativi ad essi.
2. **Negozio:** il quale raccoglie i dati riguardanti i negozi presenti nella nostra app, esso contiene una sottoschema per raccogliere gli orari del negozio.
3. **Ecommerce:** che raccoglie tutte le informazioni sugli utenti che vendono attraverso e-commerce e sono disposte a vendere oggetti fisicamente nella città di Trento.
4. **Feedback:** raccoglie i feedback positivi o negativi riguardanti i negozi non ancora verificati, in questo modello abbiamo creato un index per far sì che un utente possa lasciare un solo feedback per negozio.
5. **MessaggioPromozionale:** il quale raccoglie i dati relativi ai messaggi promozionali scritti dai vendori per gli utenti, esso raccoglie anche i messaggi di sistema.
6. **Recensione:** raccoglie le recensioni che gli utenti possono lasciare ai vari negozi.

7. **SegnalazioneErroriDati:** raccoglie le segnalazioni di errori nei dati riguardanti un negozio da parte degli utenti. Gli errori sono divisi in 4 categorie che distinguiamo grazie al campo "motivo" nel modello ("attività chiusa", "mancato rispetto sostenibilità", "dati errati", "altro").

### 3.5. Testing

N. Test Case	Descrizione Test Case	Test Data	Precondizioni	Dipendenze	Risultato Atteso	Risultato riscontrato	Note
1	Autenticazione corretta POST /api/auth/login	username:validUsername password:validPassword	le credenziali devono essere valide	-	Viene dato l'accesso all'utente, il sistema restituisce 200	Viene dato l'accesso all'utente, il sistema restituisce 200	-
1.1	Autenticazione senza username POST /api/auth/login	username vuota password:validPassword	-	-	Viene negato l'accesso all'utente, Error 400 e un messaggio di errore <Bad Request>	Viene negato l'accesso all'utente, Error 400 e un messaggio di errore <Bad Request>	-
1.2	Autenticazione senza password POST /api/auth/login	username:validUsername password vuota	-	-	Viene negato l'accesso all'utente, Error 400 e un messaggio di errore <Bad Request>	Viene negato l'accesso all'utente, Error 400 e un messaggio di errore <Bad Request>	-
1.3	Autenticazione con username non corretto (ovvero l'utente non viene trovato) POST /api/auth/login	username:invalidUsername, password:validPassword	-	-	Viene negato l'accesso all'utente, Error 401 e un messaggio di errore <Unauthorized>	Viene negato l'accesso all'utente, Error 401 e un messaggio di errore <Unauthorized>	-
1.4	Autenticazione con password errata POST /api/auth/login	username:validUsername, password:invalidPassword	-	-	Viene negato l'accesso all'utente, Error 401 e un messaggio di errore <Unauthorized>	Viene negato l'accesso all'utente, Error 401 e un messaggio di errore <Unauthorized>	-
1.5	Autenticazione se il server fallisce	-	-	-	Viene negato l'accesso	Viene negato l'accesso	Errore DB

	nel stabilire una connessione con il database POST /api/auth/login				all'utente, Error 500 e un messaggio di errore <Internal Server Error>	all'utente, Error 500 e un messaggio di errore <Internal Server Error>	simulato tramite mock.
2	Ricerca di negozi corretta GET /api/negozi	per utente autenticato JWT tokenUtente Generico	per utente autenticato JWT valido	-	il sistema restituisce 200 e un array con i negozi corrispondenti alla ricerca (solo negozi con VerificatoDaOperatore == true)	restituisce 200 e un array con i negozi corrispondenti alla ricerca (solo negozi con VerificatoDaOperatore == true)	Per utenti autenticati e non
2.1	Ricerca di negozi corretta (operatore) GET /api/negozi	Header: Authorization: Bearer tokenOperatore	Operatore autenticato (JWT valido)	-	il sistema restituisce 200 e un array con i negozi corrispondenti alla ricerca (inclusi quelli non verificati)	restituisce 200 e un array con i negozi corrispondenti alla ricerca (inclusi quelli non verificati)	Operatore vede anche verificatoDaOperatore: false.
3	Ricerca negozio corretto (utente) GET /api/negozi/{negozio_id}	Path: {negozi_id} =IDNegozio Header: Authorization: Bearer tokenUtente Generico	Utente autenticato (JWT valido), negozio esistente IDNegozio	-	il sistema restituisce 200 e le informazioni del negozio selezionato	restituisce 200 e le informazioni del negozio selezionato	Per utenti autenticati e non, alcuni campi non sono visibili: licenza Oppure Foto, proprietario, verificatoDaOperatore.
3.1	Ricerca negozio corretto (proprietario) GET /api/negozi/{negozio_id}	Path: {negozi_id} =IDNegozio Header: Authorization: Bearer tokenProprietario	Proprietario autenticato (JWT valido). Negozio esistente IDNegozio.	-	il sistema restituisce 200 e le informazioni del negozio selezionato.	restituisce 200 e le informazioni del negozio selezionato.	Campi visibili includono licenza Oppure Foto e proprietario
3.2	Ricerca negozio corretto(operator e) GET	Path: {negozi_id} =IDNegozio	Operatore autenticato (JWT valido). Negozio	-	il sistema restituisce 200 e le informazioni	restituisce 200 e le informazioni	Campi visibili includono

	/api/negozi/{negozio_id}	Header: Authorization: Bearer tokenOperatore	esistente IDNegozio.		del negozio selezionato.	del negozio selezionato.	licenza Oppure Foto, verificataDaOperatore, proprietarioInAttesa.
3.3	Ricerca di un negozio non esistente GET /api/negozi/{negozio_id}	Path: {negozio_id} =FakeIDNegozio Header: Authorization: Bearer tokenUtente Generico	utente autenticato JWT valido	-	Viene negato il recupero del negozio, Error 404 e un messaggio di errore <NotFound>	Viene negato il recupero del negozio, Error 404 e un messaggio di errore <NotFound>	-
3.4	Ricerca di un negozio se il server fallisce nel stabilire una connessione con il database GET /api/negozi/{negozio_id}	-	-	-	Viene negato il recupero, Error 500 e un messaggio di errore <Internal Server Error>	Viene negato il recupero, Error 500 e un messaggio di errore <Internal Server Error>	Errore DB simulato tramite mock.
4	Creazione negozio corretta (proprietario) POST /api/negozi	Header: Authorization: Bearer tokenProprietario Body: payloadValido	proprietario autenticato Payload valido	-	Viene creato il negozio, il sistema restituisce 201	Viene creato il negozio, il sistema restituisce 201	-
4.1	Creazione negozio con campi obbligatori mancanti POST /api/negozi	Header: Authorization: Bearer tokenProprietario Body: { negozio senza una categoria scelta }	Proprietario autenticato Payload incompleto	-	Viene negata la creazione del negozio, Error 400 e un messaggio di errore <Bad Request>	Viene negata la creazione del negozio, Error 400 e un messaggio di errore <Bad Request>	la categoria del negozio è obbligatoria
4.2	Creazione negozio con token scaduto/non valido POST /api/negozi	Header: Authorization: Bearer expiredToken Body: payloadValido	Token scaduto.	-	Viene negata la creazione del negozio, Error 401 e un messaggio di errore <Unauthorized>	Viene negata la creazione del negozio, Error 401 e un messaggio di errore <Unauthorized>	-

4.3	Creazione negozio con token che non fornisce i permessi richiesti POST /api/negozi	Header: Authorization: Bearer FakelDUser (token senza permessi richiesti) Body: payloadValido	token che non fornisce i permessi richiesti	-	Viene negata la creazione del negozio, Error 403 un messaggio di errore <Forbidden>	Viene negata la creazione del negozio, Error 403 un messaggio di errore <Forbidden>	Nel test viene passato un ObjectId come "token" per simular e un token senza permessi richiesti .
4.4	Creazione negozio se il server fallisce nel stabilire una connessione con il database POST /api/negozi	-	-	-	Viene negata la creazione, Error 500 e un messaggio di errore <Internal Server Error>	Viene negata la creazione, Error 500 e un messaggio di errore <Internal Server Error>	Errore DB simulato tramite mock.
5	eliminazione negozio da utente DELETE /api/negozi/{negozio_id}	Path: {negozi_id} =negoziDa EliminareId Header: Authorization: Bearer tokenUtente Generico	Utente autenticato ma senza permessi. Esiste "NegozioDaEliminare" creato in beforeAll	-	Viene negata l'eliminazione, Error 403 e un messaggio di errore <Forbidden>	Viene negata l'eliminazione, Error 403 e un messaggio di errore <Forbidden>	-
5.1	eliminazione negozio con token scaduto DELETE /api/negozi/{negozi_id}	Path: {negozi_id} =negoziDa EliminareId Header: Authorization: Bearer expiredToken	Token scaduto. Negozio esistente.	-	Viene negata l'eliminazione, Error 401 e un messaggio di errore <Unauthorized>	Viene negata l'eliminazione, Error 401 e un messaggio di errore <Unauthorized>	-
5.2	eliminazione negozio se il server fallisce nel stabilire una connessione con il database DELETE /api/negozi/{negozio_id}	-	-	-	Viene negata l'eliminazione, Error 500 e un messaggio di errore <Internal Server Error>	Viene negata l'eliminazione, Error 500 e un messaggio di errore <Internal Server Error>	Errore DB simulato tramite mock.
5.3	eliminazione corretta (operatore) DELETE	Path: {negozi_id} =negoziDa EliminareId	Operatore autenticato. Negozio esistente.	va fatto dopo tutti i test fallimenti	Viene eliminato il negozio, il sistema restituisce 200	Viene eliminato il negozio, il sistema	Verifica: findById ritorna null.

	/api/negozi/{negozio_id}	Header: Authorization: Bearer tokenOperatore		Se non elimina già il negozi che i test precede nti vorrebbe ro eliminar e (se no l'errore verrà sempre negozi non esistent e)		restituisce 200	
5.4	eliminazione di un negozi non esiste DELETE /api/negozi/{negozio_id}	Path: {negozi_id} =FakeIDNegozio Header: Authorization: Bearer tokenOperatore	Operatore autenticato. ID negozi inesistente.	-	Viene negata l'eliminazione, Error 404 e un messaggio di errore <NotFound>	Viene negata l'eliminazione, Error 404 e un messaggio di errore <NotFound>	-
6	Aggiornamento negozi da utente senza permessi PUT /api/negozi/{negozi_id}	Path: {negozi_id} =negoziold Header: Authorization: Bearer tokenUtente Generico  Body: payloadModificaProprietario (modifiche che vorrebbe approntare)	Esiste "NegoziDaModificare". Utente autenticato ma senza permessi di modifica.	-	Viene negata la modifica, Error 403 e un messaggio di errore <Forbidden>	Viene negata la modifica, Error 403 e un messaggio di errore <Forbidden>	-
6.1	Aggiornamento negozi con token scaduto PUT /api/negozi/{negozi_id}	Path: {negozi_id} =negoziold Header: Authorization: Bearer expiredToken  Body: payloadModificaProprietario	Esiste "NegoziDaModificare". Token scaduto.		Viene negata la modifica, Error 401 e un messaggio di errore <Unauthorized>	Viene negata la modifica, Error 401 e <Unauthorized>	-

6.2	Aggiornamento negozio se il server fallisce nel stabilire una connessione con il database PUT /api/negozi/{negozi_id}	-	-	-	Viene negata la modifica, Error 500 e un messaggio di errore <Internal Server Error>	Viene negata la modifica, Error 500 e un messaggio di errore <Internal Server Error>	Errore DB simulato tramite mock.
6.3	Aggiornamento negozio corretto (da operatore) PUT /api/negozi/{negozi_id}	Path: {negozi_id} =negoziold  Header: Authorization: Bearer tokenOperatore  Body: payloadModificaOperatore	Esiste "NegozioDaModificare". Operatore autenticato.	-	Viene modificato il negozio, il sistema restituisce 200	Viene modificato il negozio, il sistema restituisce 200	-
6.4	Aggiornamento negozio corretto(proprietario) PUT /api/negozi/{negozi_id}	Path: {negozi_id} =negoziold  Header: Authorization: Bearer tokenProprietario  Body: payloadModificaProprietario	Esiste "NegozioDaModificare". Venditore autenticato.	-	Viene modificato il negozio, il sistema restituisce 200	Viene modificato il negozio, il sistema restituisce 200	-
6.5	Aggiornamento negozio non esiste PUT /api/negozi/{negozi_id}	Path: {negozi_id} =FakeIDNegozio  Header: Authorization: Bearer tokenOperatore  Body: payloadModificaOperatore	Non esiste "NegozioDaModificare". Operatore autenticato.	-	Viene negata la modifica, Error 404 e un messaggio di errore <NotFound>	Viene negata la modifica, Error 404 e un messaggio di errore <NotFound>	-
6.6	Aggiornamento negozio con campi del payload	Path: {negozi_id} =negoziold	Operatore autenticato. Payload non valido.	-	Viene negata la modifica, Error 400 e un messaggio di	Viene negata la modifica, Error 400 e un	-

	mancanti/non validi PUT /api/negozi/{negozi_id}	Header: Authorization: Bearer tokenOperatore Body: { nome:"" }(il nome non può essere una stringa vuota)			errore <Bad Request>	messaggio di errore <Bad Request>	
7	Visualizzazione preferiti corretta  GET /api/preferiti/users/{user_id}	Path: {user_id}=ID Utente Header: Authorization: Bearer tokenUtente	Utente autenticato (JWT valido)	-	il sistema restituisce 200 e un array di negozi preferiti.	il sistema restituisce 200 e un array di negozi preferiti.	-
7.1	Visualizzazione preferiti di un utente da parte di un altro utente  GET /api/preferiti/users/{user_id}	Path: {user_id}=ID Utente Header: Authorization: Bearer tokenVenditore	Richiedente autenticato ma non corrisponde a {user_id} (utente diverso).	-	Viene negato l'accesso, Error 403 e un messaggio di errore <Forbidden>	Viene negato l'accesso, Error 403 e un messaggio di errore <Forbidden>	-
7.2	Visualizzazione preferiti con token non valido o scaduto  GET /api/preferiti/users/{user_id}	Path: {user_id}=ID Utente Header: Authorization: Bearer expiredToken	Token scaduto.	-	Viene negato l'accesso, Error 401 e un messaggio di errore <Unauthorized>	Viene negato l'accesso, Error 401 e un messaggio di errore <Unauthorized>	-
7.3	Visualizzazione preferiti da parte di un Utente non esiste  GET /api/preferiti/users/{user_id}	Path: {user_id}=FakeIDUtente Header: Authorization: Bearer token firmato con id=FakeIDUtente	Richiesta autenticata ma l'utente non è presente in database.	-	Viene negato l'accesso, Error 404 e un messaggio di errore <NotFound>	Viene negato l'accesso, Error 404 e un messaggio di errore <NotFound>	-
7.4	Visualizzazione preferiti se il server fallisce nel stabilire una connessione con il database	-	-	-	viene negato l'accesso, Error 500 e un messaggio di errore <Internal Server Error>	viene negato l'accesso, Error 500 e un messaggio di errore <Internal Server Error>	Errore DB simulato tramite mock.
8	Aggiunta negozio ai preferiti con token scaduto	Path: {user_id}=ID Utente	Token scaduto.	-	Viene negata l'operazione, Error 401 e un	Viene negata l'operazione, Error 401 e un	-

	POST /api/preferiti/users/{user_id}	Header: Authorization: Bearer expiredToken Body: { negozio_id: IDNegozio }			messaggio di errore <Unauthorized>	un messaggio di errore <Unauthorized>	
8.1	Aggiunta negozio ai preferiti di un utente da parte di un altro utente  POST /api/preferiti/users/{user_id}	Path: {user_id}=ID Venditore Header: Authorization: Bearer tokenUtente Body: { negozio_id: IDNegozio }	Richiedente autenticato ma non corrisponde a {user_id} (utente diverso).	-	Viene negata l'operazione, Error 403 e un messaggio di errore <Forbidden>	Viene negata l'operazione, Error 403 e un messaggio di errore <Forbidden>	-
8.2	Aggiunta negozio ai preferiti da parte di un utente che non esiste  POST /api/preferiti/users/{user_id}	Path: {user_id}=Fake Utente Header: Authorization: Bearer tokenUtente Fake Body: { negozio_id: IDNegozio }	Utente non presente in database	-	Viene negata l'operazione, Error 404 e un messaggio di errore <NotFound>	Viene negata l'operazione, Error 404 e un messaggio di errore <NotFound>	-
8.3	Aggiunta negozio ai preferiti corretta  POST /api/preferiti/users/{user_id}	Path: {user_id}=ID Utente Header: Authorization: Bearer tokenUtente Body: { negozio_id: IDNegozio }	Utente esistente ed autenticato. negozio_id valido.	-	Viene aggiunto il negozio ai preferiti, il sistema restituisce 201	Viene aggiunto il negozio ai preferiti, il sistema restituisce 201	-
8.4	Aggiunta negozio ai preferiti se il server fallisce nel stabilire una connessione con il database	-	-	-	Viene negata l'operazione, Error 500 e un messaggio di errore <Internal Server Error>	Viene negata l'operazione, Error 500 e un messaggio di errore <Internal Server Error>	Errore DB simulato tramite mock.
9	Rimozione negozio dai preferiti corretta  DELETE /api/preferiti/users/{user_id}	Path: {user_id}=ID Utente Query: ?negozio_id =IDNegozio Header: Authorization: Bearer tokenUtente	Utente autenticato. negozio_id presente nei preferiti dell'utente.	-	Viene rimosso il negozio dai preferiti, il sistema restituisce 204	Viene rimosso il negozio dai preferiti, il sistema restituisce 204	-

9.1	Rimozione negozio dai preferiti in cui manca negozio_id DELETE /api/preferiti/users/{user_id}	Path: {user_id}=ID Utente Header: Authorization: Bearer tokenUtente	Utente autenticato. Parametro query mancante.	-	Viene negata l'operazione, Error 400 e un messaggio di errore <Bad Request>	Viene negata l'operazione, Error 400 e un messaggio di errore <Bad Request>	-
9.2	Rimozione negozio dai preferiti con token scaduto DELETE /api/preferiti/users/{user_id}	Path: {user_id}=ID Utente Query: ?negozioid =IDNegozio Header: Authorization: Bearer expiredToken	Token scaduto.	-	Viene negata l'operazione, Error 401 e un messaggio di errore <Unauthorized>	Viene negata l'operazione, Error 401 e un messaggio di errore <Unauthorized>	-
9.3	Rimozione negozio dai preferiti di un utente da parte di un altro utente DELETE /api/preferiti/users/{user_id}	Path: {user_id}=ID Utente Query: ?negozioid =IDNegozio Header: Authorization: Bearer tokenVenditore	Richiedente autenticato ma non corrisponde a {user_id} (utente diverso).	-	Viene negata l'operazione, Error 403 e un messaggio di errore <Forbidden>	Viene negata l'operazione, Error 403 e un messaggio di errore <Forbidden>	-
9.4	Rimozione negozio dai preferiti da parte di un utente non esiste DELETE /api/preferiti/users/{user_id}	Path: {user_id}=Fake Utente Query: ?negozioid =IDNegozio Header: Authorization: Bearer tokenUtente Fake	Utente non presente nel datanase.	-	Viene negata l'operazione, Error 404 e un messaggio di errore <Not Found>	Viene negata l'operazione, Error 404 e un messaggio di errore <Not Found>	-
9.5	Rimozione negozio dai preferiti se il server fallisce nel stabilire una connessione con il database	-	-	-	Viene negata l'operazione, Error 500 e un messaggio di errore <Internal Server Error>	Viene negata l'operazione, Error 500 e un messaggio di errore <Internal Server Error>	Errore DB simulato tramite mock.
10	Ricerca feedback corretta GET /api/feedback	Query: ?negozioid =IDNegozio Header: Authorization: Bearer tokenUtente	Utente autenticato (JWT valido). Esiste il feedback dell'utente	-	Viene restituito 200 e array con il feedback dell'utente.	Viene restituito 200 e array con il feedback dell'utente.	-

			associato a IDNegozio.				
10.1	Ricerca feedback senza negozio_id nella query GET /api/feedback	Header: Authorization: Bearer tokenUtente (senza query)	Utente autenticato. Query mancante.	-	Viene negata la richiesta, Error 400 e un messaggio di errore <Bad Request>	Viene negata la richiesta, Error 400 e un messaggio di errore <Bad Request>	-
10.2	Ricerca feedback di un utente che non ha lasciato il feedback per il negozio GET /api/feedback	Query: ?negozio_id =FakeIDNegozio Header: Authorization: Bearer tokenUtente	Utente autenticato. Nessun feedback presente per la coppia (utente, negozio).	-	Viene negata la richiesta, Error 404 e un messaggio di errore <Not Found>	Viene negata la richiesta, Error 404 e un messaggio di errore <Not Found>	-
10.3	Ricerca feedback di un utente con token scaduto GET /api/feedback	Query: ?negozio_id =IDNegozio Header: Authorization: Bearer expiredToken	Token scaduto.	-	Viene negato l'accesso, Error 401 e un messaggio di errore <Unauthorized>	Viene negato l'accesso, Error 401 e un messaggio di errore <Unauthorized>	-
10.4	Ricerca feedback con un token senza i permessi per farlo GET /api/feedback	Query: ?negozio_id =IDNegozio Header: Authorization: Bearer FakeIDUtente	Token non valido (viene passato un ObjectId come token).	-	Viene negato l'accesso, Error 403 e un messaggio di errore <Forbidden>	Viene negato l'accesso, Error 403 e un messaggio di errore <Forbidden>	-
10.5	Ricerca feedback se il server fallisce nel stabilire una connessione con il database	-	-	-	Viene negata l'operazione, Error 500 e un messaggio di errore <Internal Server Error>	Viene negata l'operazione, Error 500 e un messaggio di errore <Internal Server Error>	Errore DB simulato tramite mock.
11	Creazione feedback corretta POST /api/feedback	Header: Authorization: Bearer tokenUtente Body: { negozio: IDNegozio, feedback: true}	Utente autenticato. Non esiste già feedback per (autore=IDUtente, negozio=IDNegozio) (il test fa deleteOne prima).	-	Viene creato il feedback: il sistema restituisce 201	Viene creato il feedback: il sistema restituisce 201	-
11.1	Creazione feedback già presente (ovvero	Header: Authorization: Bearer	Utente autenticato. Esiste già	il test utilizza il feedbac	Viene negata la creazione, Error 409 e un	Viene negata la creazione, Error 409 e	-

	I'utente ha già creato un feedback) POST /api/feedback	tokenUtente Body: { negozio: IDNegozio, feedback: true }	feedback per (autore=IDUtente, negozio=IDNegozio)	k inserito nel test 11 ma se per errore non è presente il test lo crea	messaggio di errore <Feedback già presente>	un messaggio di errore <Feedback già presente>	
11.2	Creazione feedback con token scaduto POST /api/feedback	Header: Authorization: Bearer expiredToken Body: { negozio: IDNegozio, feedback: true }	Token scaduto.	-	Viene negata la creazione, Error 401 e un messaggio di errore <Unauthorized>	Viene negata la creazione, Error 401 e un messaggio di errore <Unauthorized>	-
11.3	Creazione feedback con un token senza i permessi per farlo POST /api/feedback	Header: Authorization: Bearer FakelDUser Body: { negozio: IDNegozio, feedback: true }	Token non valido (ObjectId passato come token).	-	Viene negata la creazione, Error 403 e un messaggio di errore <Forbidden>	Viene negata la creazione, Error 403 e un messaggio di errore <Forbidden>	-
11.4	Creazione feedback se il server fallisce nel stabilire una connessione con il database	-	-	-	Viene negata l'operazione, Error 500 e un messaggio di errore <Internal Server Error>	Viene negata l'operazione, Error 500 e un messaggio di errore <Internal Server Error>	Errore DB simulato tramite mock.
12	Ricerca ecommerce corretta GET /api/ecommerce	-			Viene restituito 200 e un array di vendori ecommerce.	Viene restituito 200 e un array di vendori ecommerce.	-
12.1	Ricerca ecommerce filtrata per zona GET /api/ecommerce?Zone=...	Query: ?Zone=Gardolo	Esistono vendori con Zone che include "Gardolo" (altrimenti array vuoto).	-	Viene restituito 200 e un array di vendori corrispondenti, tali che Zone contiene "Gardolo".	Viene restituito 200 e un array di vendori corrispondenti, tali che Zone contiene "Gardolo".	-
12.2	Ricerca ecommerce se il server fallisce nel stabilire una	-	-	-	Viene negata l'operazione, Error 500 e un messaggio di	Viene negata l'operazione, Error 500 e un	Errore DB simulato

	connessione con il database				errore <Internal Server Error>	messaggio di errore <Internal Server Error>	tramite mock.
13	Ricerca venditore ecommerce attraverso ID corretto GET /api/ecommerce/{ecommerce_id}	Path: {ecommerce_id}=IDEcommerce	IDEcommerce esiste	-	Viene restituito 200 e i dettagli del venditore.	Viene restituito 200 e i dettagli del venditore.	-
13.1	Ricerca venditore ecommerce attraverso ID se il server fallisce nel stabilire una connessione con il database	-	-	-	Viene negata l'operazione, Error 500 e un messaggio di errore <Internal Server Error>	Viene negata l'operazione, Error 500 e un messaggio di errore <Internal Server Error>	Errore DB simulato tramite mock.
13.2	Ricerca venditore ecommerce attraverso ID valido ma il venditore associato è inesistente GET /api/ecommerce/{ecommerce_id}	Path: {ecommerce_id}=idInesistente (ObjectId)	Nessun venditore con quell'ID.	-	Viene negata la creazione, Error 404 e un messaggio di errore <Not Found>	Viene negata la creazione, Error 404 e un messaggio di errore <Not Found>	-
13.3	Ricerca venditore ecommerce attraverso ID con formato non valido GET /api/ecommerce/{ecommerce_id}	Path: {ecommerce_id}="id-non-valido"	-	-	Viene negata la creazione, Error 400 e un messaggio di errore <Bad Request>	Viene negata la creazione, Error 400 e un messaggio di errore <Bad Request>	-

## 4. Front End

Il frontend fornisce le funzionalità di ricerca, visualizzazione, inserimento e cancellazione dei dati dell'applicazione. La WebApp è composta dalle seguenti pagine:

- Schermata home per la ricerca di un'attività commerciale;
- Schermata di login
- Risultato selezione segnaposto
- Schermata vendori E-commerce
- Risultato selezione di una zona E-commerce
- Selezione di un venditore E-commerce
- Schermata preferiti
- Schermata notifiche

Document:

Descrizione di Progetto

Revision:

1.0

- Schermata segnalazione attività altrui
- Schermata segnalazione/rivendicazione propria attività

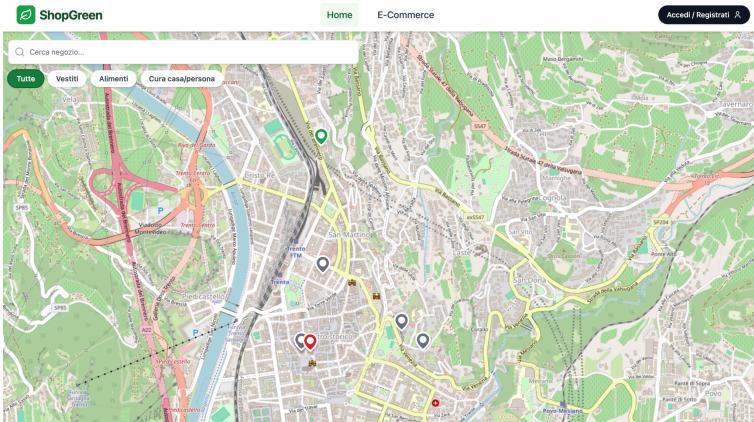


Figura 4.1 – Schermata home per la ricerca di un’attività commerciale

In figura 4.1 è mostrata la home page dell’applicazione. Da questa schermata l’utente può ricercare un’attività commerciale, selezionare un segnaposto per visualizzare i dettagli di un negozio, accedere alla sezione dedicata ai vendori e-commerce oppure alla sezione del login. Inoltre, se l’utente ha effettuato l’accesso, può accedere alla sezione dei preferiti e segnalare un’attività.

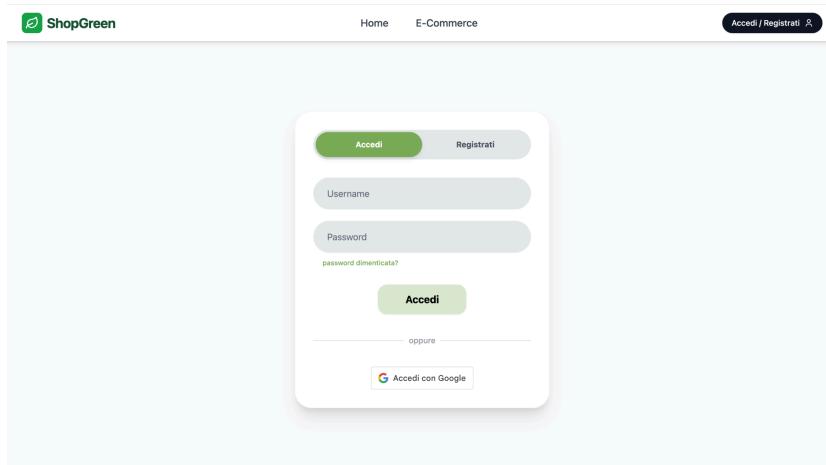


Figura 4.2 – Schermata di login

In figura 4.2 è mostrata la schermata per effettuare l’accesso alla WebApp. L’utente può inserire le credenziali scelte in fase di registrazione oppure accedere mediante Google.

Document:

Descrizione di Progetto

Revision:

1.0

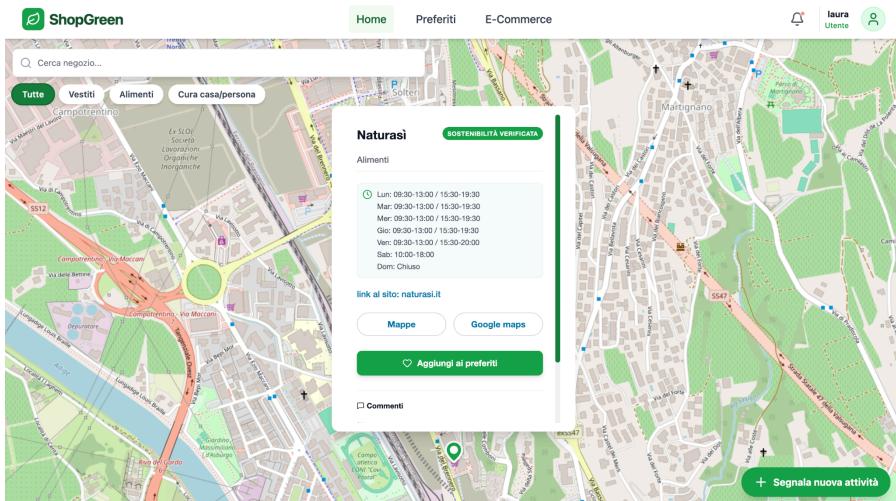


Figura 4.3 – Risultato selezione segnaposto

In figura 4.3 è mostrato il pop-up che appare a seguito della selezione di un segnaposto nella schermata home. L’utente può visualizzare i dettagli di un’attività commerciale, quali nome, categoria, orari di apertura e link utili (sito, Google Maps, Mappe di ios). Se l’utente ha effettuato l’accesso, può aggiungere l’attività selezionata ai preferiti e, se questa non è verificata secondo i criteri di sostenibilità, può votare al sondaggio apposito.

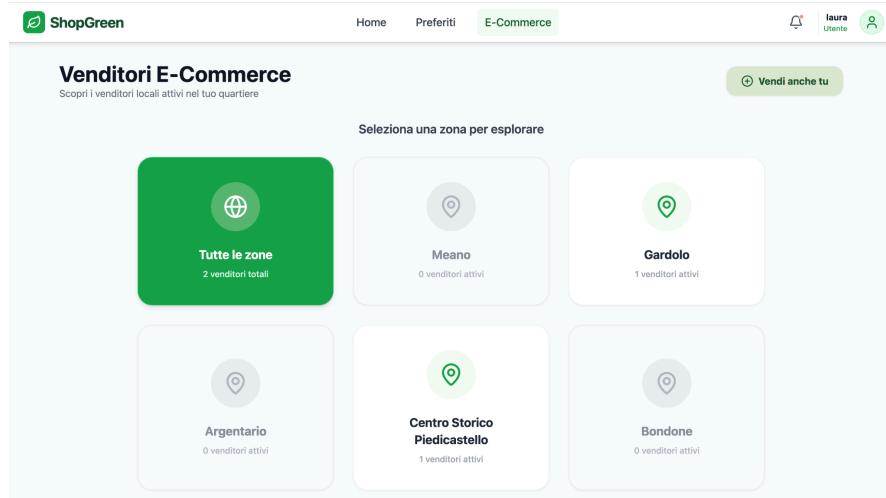


Figura 4.4 – Schermata venditori e-commerce

In figura 4.4 è mostrata la sezione dedicata ai venditori di e-commerce. L’utente può visualizzare le zone della città di Trento e selezionarne una per accedere alla lista dei venditori disposti ad incontrarsi con i clienti in tale zona.

Document:

Descrizione di Progetto

Revision:

1.0

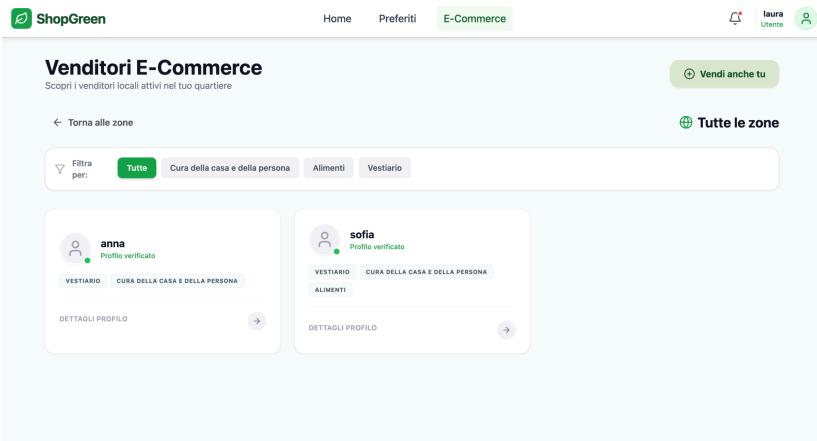


Figura 4.5 – Risultato selezione zona e-commerce

In figura 4.5 è mostrata la schermata a cui viene indirizzato l'utente una volta selezionata la zona di cui intende visualizzare i venditori attivi. L'utente può filtrare i venditori per categoria di prodotti venduti e visualizzare così una lista di questi.

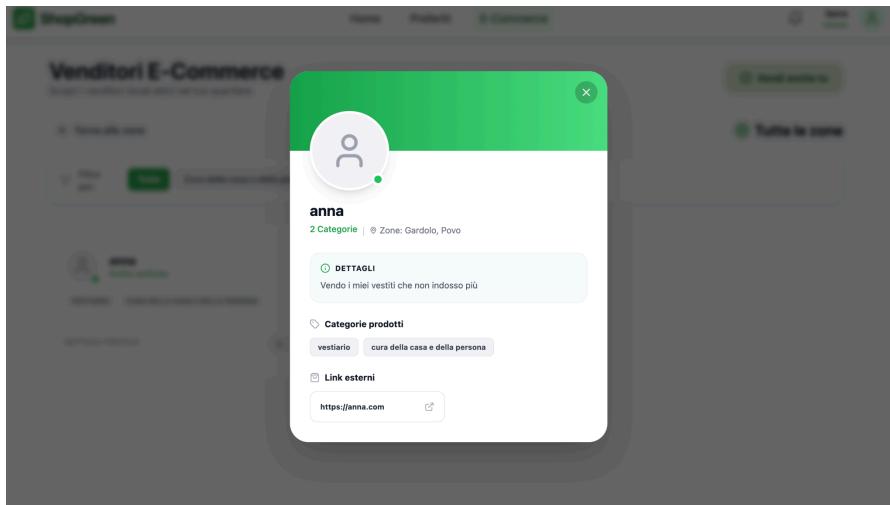


Figura 4.6 – Risultato selezione venditori e-commerce

In figura 4.6 è mostrato il pop-up che appare a seguito della selezione di un venditore e-commerce. L'utente può visualizzare i dettagli del venditore, quali nome, zone della città di Trento in cui è disposto ad incontrarsi con i clienti, categorie di prodotti venduti, link ai siti di vendita ed un'eventuale descrizione.



Document:

Descrizione di Progetto

Revision:

1.0

Figura 4.7 – Schermata preferiti

In figura 4.7 è mostrata la sezione dedicata ai preferiti di un utente autenticato. Quest'ultimo può visualizzare una lista delle attività commerciali aggiunte tramite il pop-up di figura 4.3. Inoltre, può selezionare un'attività per essere reindirizzato al negozio nella schermata home oppure eliminare la stessa dai preferiti.

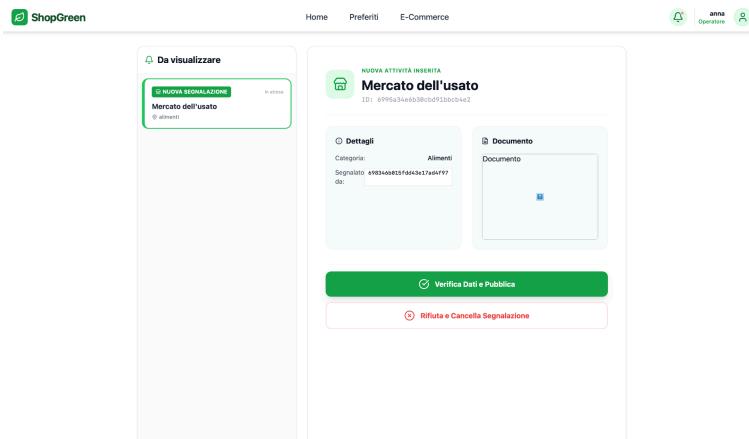


Figura 4.8 – Schermata notifiche

In figura 4.8 è mostrata la sezione dedicata alle notifiche, presente solo per gli utenti autenticati. L'utente può consultare la lista delle notifiche nella barra a sinistra e selezionarne una per visualizzarne un'anteprima.

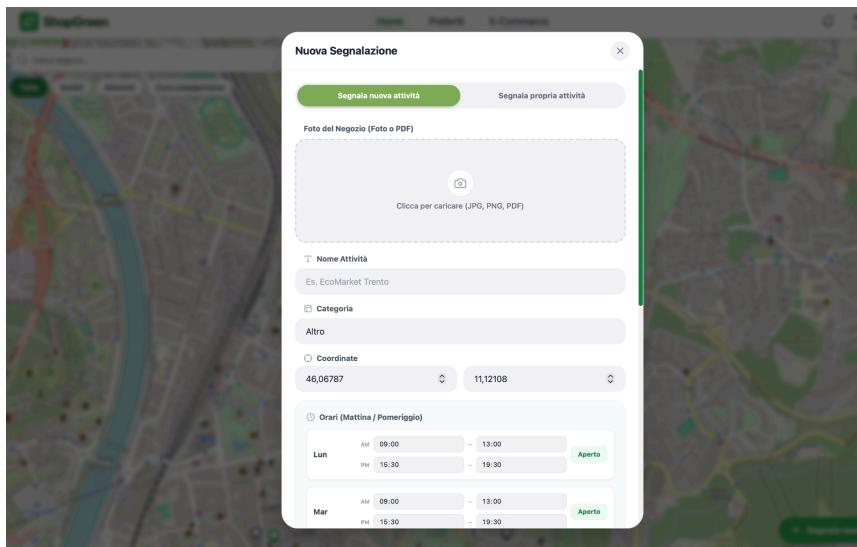


Figura 4.9 – Schermata segnalazione attività altrui

In figura 4.9 è mostrata la finestra che appare a seguito della selezione del pulsante “Segnala attività” presente nella schermata home e della sezione “Segnala nuova attività”. L’utente autenticato può segnalare un’attività altrui inserendone l’immagine, il nome, la categoria di prodotti venduti, le coordinate, gli orari e link utili.

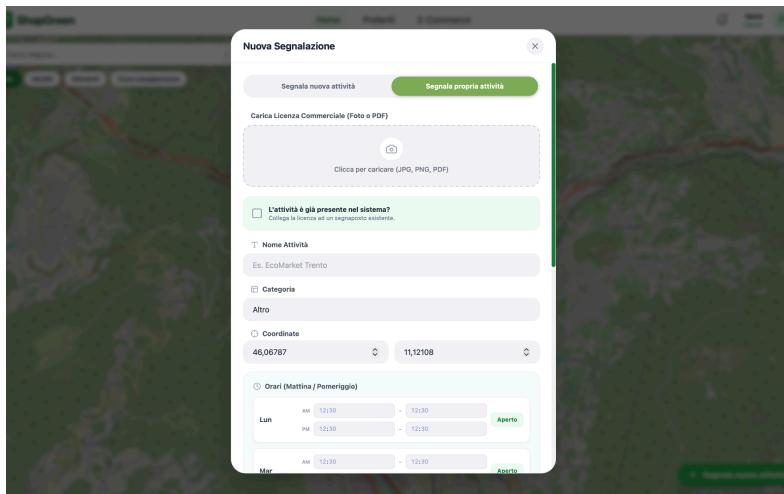


Figura 4.10 – Schermata segnalazione/rivendicazione propria attività

In figura 4.10 è mostrata la finestra che appare a seguito della selezione del pulsante “Segnala attività” presente nella schermata home e della sezione “Segnala propria attività”. L’utente autenticato e proprietario di un’attività non ancora presente può inserire i dati nel form, quali licenza di vendita, nome, categoria di prodotti venduti, coordinate, orari di apertura e link utili. L’utente autenticato e proprietario di un’attività già presente che intende rivendicare la propria attività, può selezionare “L’attività è già presente nel sistema?” e scegliere l’attività dall’elenco, così i campi verranno riempiti in automatico, pur rimanendo modificabili.

## 5. Deployment

Backend e frontend sono deployati separatamente sulla piattaforma [render.com](https://render.com) e sono disponibili ai seguenti link:

- Backend: <https://greenshop-n17k.onrender.com>
- Frontend: <https://shopgreen-frontend.onrender.com>

Per accedere al deploy della WebApp è possibile utilizzare i seguenti account per le tre tipologie di utenti:

- Operatore
  - Username: operatoreProva
  - Password: operatore
- Utente non venditore
  - Username: utenteProva
  - Password: utente
- Utente venditore (proprietario del negozio Naturasi)

Document:

Descrizione di Progetto

Revision:

1.0

---

- Username: venditoreProva
- Password: venditore

Per problemi legati al deploy, contattare:

- [anna.luvisotto@studenti.unitn.it](mailto:anna.luvisotto@studenti.unitn.it) (amministratrice)
- [laura.prandina@studenti.unitn.it](mailto:laura.prandina@studenti.unitn.it) (rappresentante del gruppo)
- [sofia.cestari@studenti.unitn.it](mailto:sofia.cestari@studenti.unitn.it) (membro del gruppo)