Implementation Report: URL Shortener Service

1. System Overview

The URL shortener service is implemented with Python, Flask framework, sqlite and a RESTful API.

2. Detailed API Specifications

The service adheres to the following operational behavior:

Method	Endpoint	Input	Functionality	Response Codes
POST	`/`	`{"value": "<url>"}`	Generates a unique short code.	201, id; 400, "error"
GET	`/<url_id>`	URL ID	Redirects to the original URL.	301, value; 404
PUT	`/<url_id>`	`{"url": "<new_url>"}`	Updates the original URL if the new URL is valid.	200; 400, "error"; 404
DELETE	`/<url_id>`	URL ID	Deletes the URL mapping.	204; 404
GET	`/`	None	Retrieves all stored URL mappings.	200, keys; 404, None
DELETE	`/`	None	Deletes all URL mappings.	404

3. Key Design

3.1 Short URL Generation Algorithm

- **Mechanism**: Randomly generated strings of length 4 (default) or 6, using alphanumeric characters (62 possible values per character). The capacity is about 1.4M for 4-digit and 56.8B for 6-digit.
- **Collision Handling**: Database uniqueness checks ensure no duplicate short codes. If a collision occurs, regeneration is performed until a unique code is found.

3.2 URL Validation

- **Regex Pattern**:

  ```
  r'^(https?://)(([A-Za-z0-9-]+(\.[A-Za-z]{2,})+)|localhost|(\d{1,3}\.){3}\d{1,3})(:\d+)?(/[^\s]*)?$'
  ```

  Requires `http://` or `https://` prefix, valid domain/IP, optional port/path.

4. Multi-User Support

To enable user-specific URL management, the following modifications are required:

1. **User Authentication**:
    - Implement OAuth 2.0 or JWT-based authentication. Users receive a token upon login, passed via the `Authorization` header.
2. **Database Enhancement**:
    - Add a `user_id` column to the `urls` table to associate URLs with user accounts.
3. **Access Control**:
    - **Authorization Middleware**: Validate tokens and restrict operations (e.g., PUT/DELETE) to URLs owned by the authenticated user.
    - **Role-Based Permissions**: Introduce `admin` roles for global management capabilities.
4. **API Adjustments**:
    - Modify the POST endpoint to include `user_id` during URL creation.
    - Filter GET requests to return only URLs owned by the current user.

---

5. Teamwork

Name	Specific Tasks
Hongyu Chen	- Modify existing code (optimize logic, fix bugs). - Write the project report (implementation details, design decisions).
Yueming Sun	- Develop core features (short URL generation, regex URL validation). - Implement API endpoints (POST/GET/PUT/DELETE).
CY Yau Chun Yuen	- Help write project report (joining the group later)