IDG2001 - Report on Assignment 2

Basic information

- Task: Improve the Social Media Service, "Twitter from Temu", into a "First Price Twitter".
- Title: Cheeper [another bird sound, also sounds like "cheaper" which fits.]
- Group: Kaja, Jon, Stian

New technologies

Docker

We decided to move our development server over to Docker for easier development and testing.

Running project with Docker (local server)

1. Make sure an .env with database credentials are included in the root of the project directory
2. Run "docker compose up --build" in terminal
3. Run Visual Stuio Code Live Server of "index.html"

New features

Like posts batcher

While like posts was already implemented in assignment 1, they were not fully functioning. In this second iteration like posts is fully implements with like posts batcher. To prevent uneccesary load on the database likes are sent to database every 1000 likes or after 1 minute ??????

Hashtags

Hashtags are identified and extracted from post text and saved to database. They can be searched for in the nav-bar search feature.

```
################
### Hashtags ###
################

def extract_hashtags_from_text(text: str) -> list[str]:
    return list(set(re.findall(r"#\w+", text.lower())))
```

Reply to posts

Posts can now be replied to by other users. Replies are stored as new posts but has a relationship with the post being replied to.

```python
class Post(Base):
    __tablename__ = 'posts'
    id = Column(Integer, primary_key=True, index=True)
    content = Column(Text)
    created_at = Column(DateTime, default=func.now())
    edited = Column(Boolean, default=False)
    user_id = Column(Integer, ForeignKey('users.id'))
    reply_to_id = Column(Integer, ForeignKey('posts.id'), nullable=True)

    author = relationship("User", back_populates="posts")
    likes = relationship("User", secondary=likes_table, back_populates="liked_posts")
    hashtags = relationship("Hashtag", secondary=post_hashtags, back_populates="posts")
    reply_to = relationship("Post", remote_side=[id], backref="replies")
```

Login/Profile Toggle

When session cookie is created and user is authenticated "Login" changes into user's profile link.

Edit/Delete Post

Users can edit and delete posts in their profile where username, email and user's posts are displayed.

Search Bar

In this iteration of Chepper the search bar is fully functional. User is able to search for content in posts, user accounts and hashtags used in posts.

Caching

Caching provided with Redis through Docker.... More information needed DB CACHING...

Load Balancer

Logger

Logs of calls to the API end points can be accessed via endpoint get/logs and will give information on request method, path, status code and timestamp.

```
{
  "method": "OPTIONS",
  "path": "/posts/37",
  "status_code": 200,
  "timestamp": "2025-05-20T21:21:02.532283"
},
{
  "method": "DELETE",
  "path": "/posts/37",
  "status_code": 200,
  "timestamp": "2025-05-20T21:21:02.923282"
},
{
  "method": "GET",
  "path": "/users/7/posts",
  "status_code": 200,
  "timestamp": "2025-05-20T21:21:04.224073"
},
```