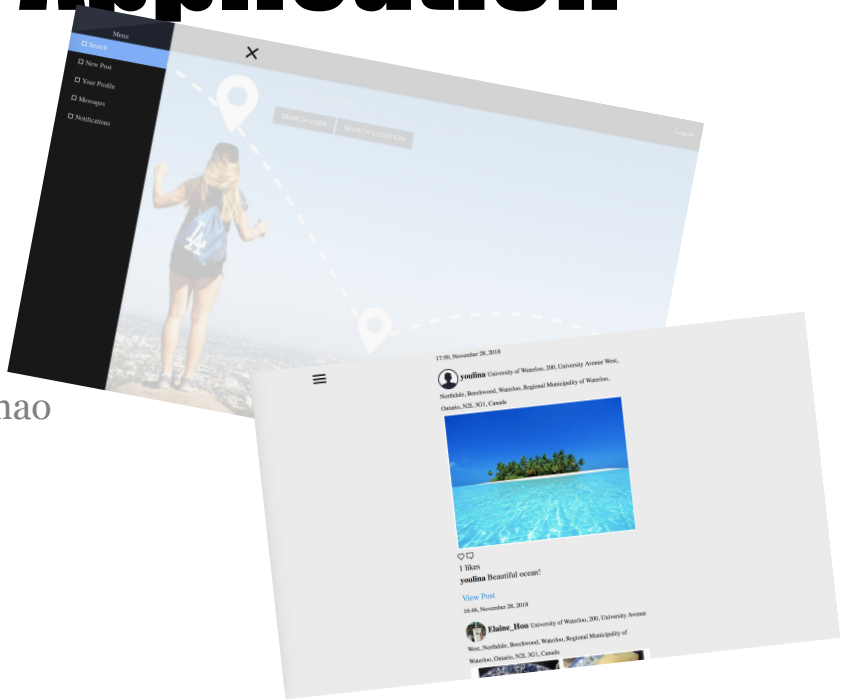
TravelFun Web Application

29/11/18



Presented by: Group 6
Tianchang Hou, Shi Cheng, Siqi Shao, Ye Wang, Yuqing Zhao

Electrical and Computer Engineering

UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING

Agenda

- Project Description
- Functional Properties
- Non-functional Properties
- Architectural Style & Design Pattern
- Conponent Diagram
- Demo Display
- Conclusion & Further Development

Project discription

- Web Application -- TravelFun
- Travel-themed social networking website
  - Worldwide travel scenes
  - Users with same interest

Functional Properties

- User Authentication
  - Register
  - Sign-in
  - Password
- Account Management
  - Profile
  - Map
- Search
  - by Location
  - by Username
- Post
  - Edit/ Upload/ Delete
  - Real-time Location
- Setting

Functional Properties cont.

- Friend Interactions
  - Follow and Unfollow
  - Like
  - Comment
  - Share
- Direct Message
- Notification

Non-Functional Properties

- Response time
  - Within 1s
  - Cache
- Useability
  - PC & Mobile devices
- Security
  - Provide access only to legitimate users (e.g sending verification email; only accepting keyboard input; CAPTCHA)
  - SSL
  - Django auth module
- Scalability
  - AWS: https://aws.amazon.com/elasticloadbalancing/pricing/
  - Apache
- Database concurrency
  - MVCC Mechanism

Architecture Style & Design Pattern

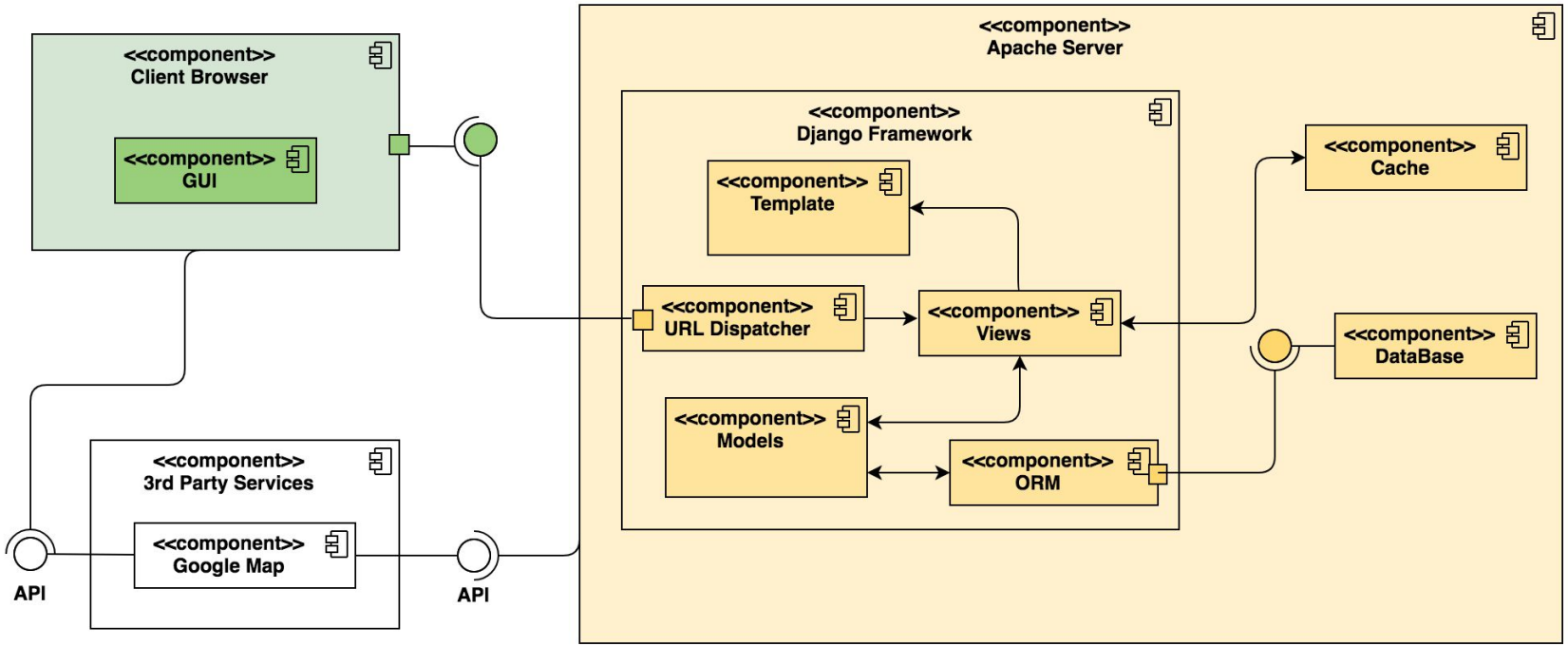
Browser-Server architrcture style:

- Browser - Web Browsers (PCs and mobile devices)
- Server - Django Web Framework and Database
- Third-party services (Google Map API for both Browser and Server)

Model-Template-View (MTV) design pattern:

- Template: provide the display logic
- Model: create data relationships and provide access to the database
- View: handles all the business logic, interacting with the model

Conponent Diagram

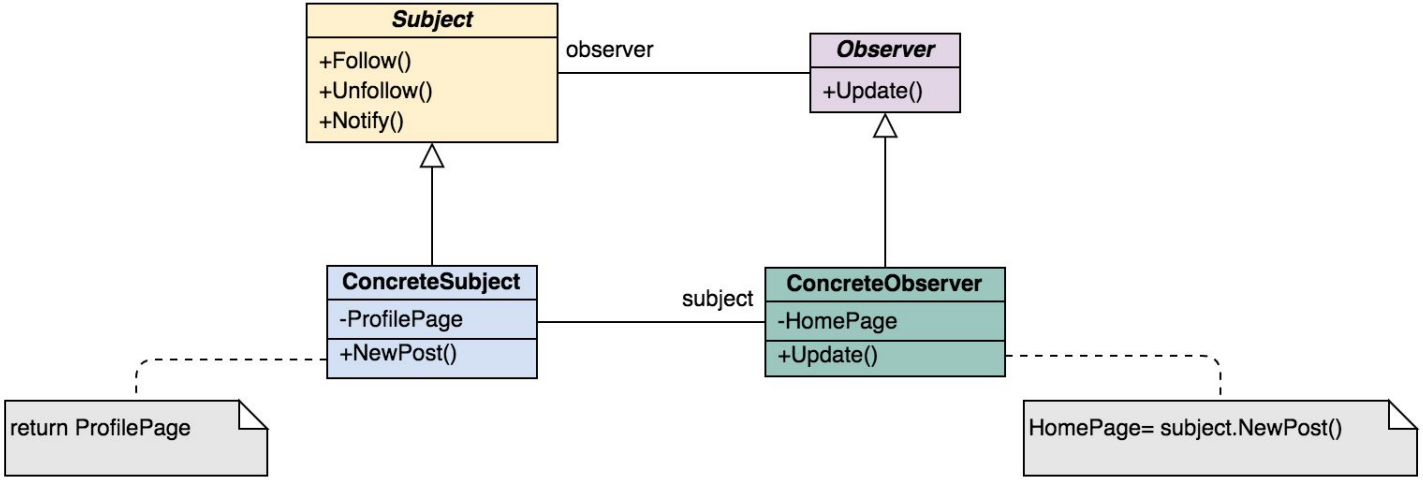


Design Pattern-Observer

Observer Pattern:

- The observer pattern establishes a one-to-many dependency between objects so that when one object changes its state, all its dependents are notified automatically.
- The observer design pattern consists of subjects and observers.

Observer Pattern

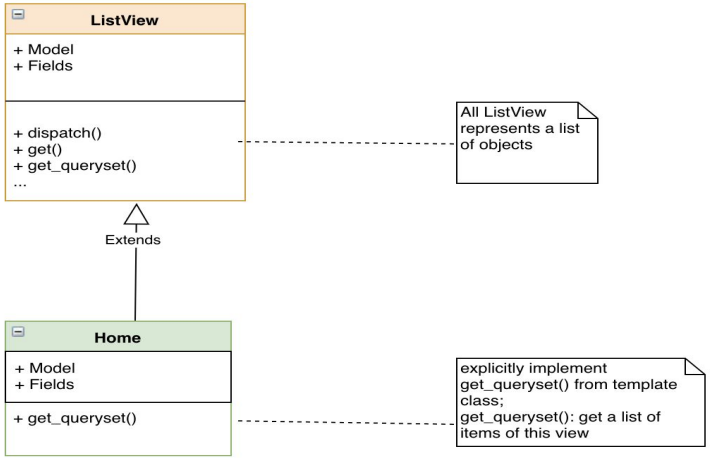


Design Pattern-Template Method

- class-based generic views (e.g CreateView, ListView...)
- override attributes and methods
- “part of the algorithm without reimplementing the whole thing”

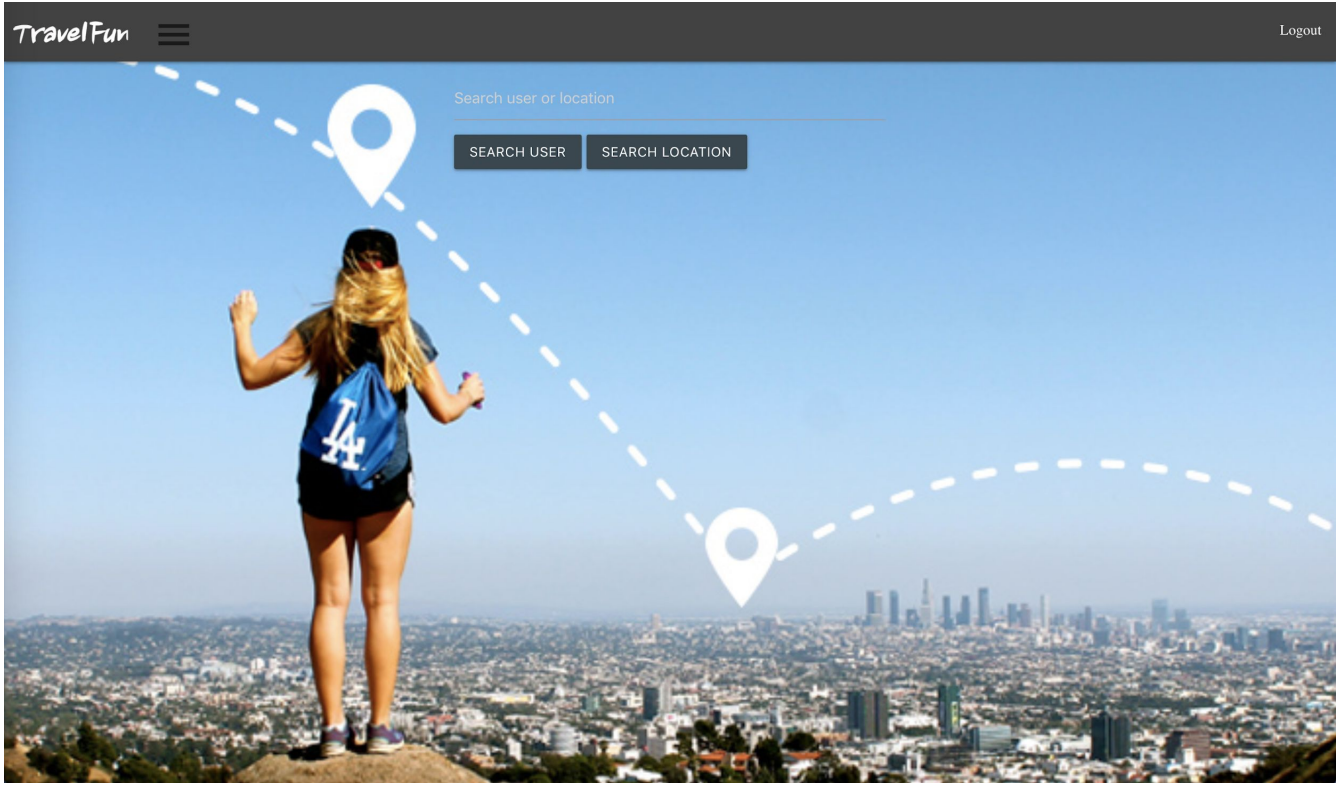
Method Flowchart

1. `dispatch()`
2. `http_method_not_allowed()`
3. `get_template_names()`
4. `get_queryset()`
5. `get_context_object_name()`
6. `get_context_data()`
7. `get()`
8. `render_to_response()`



Demo Display

https://35.183.67.139



Conclusion & Further Development

Conclusion

- Client-Server architecture style
- MTV Pattern and Observer Pattern
- Functional and Non-functional properties realization

Further Development

- Open authority in Post
- Privacy Setting in Direct Message and Notifications

Challenges

- Improve the non-functional properties, like response speed and scalability

Q & A