SWE 530: Software Design Process

MVC Design Pattern

Dr. H. Birkan YILMAZ
Department of Computer Engineering
Boğaziçi University

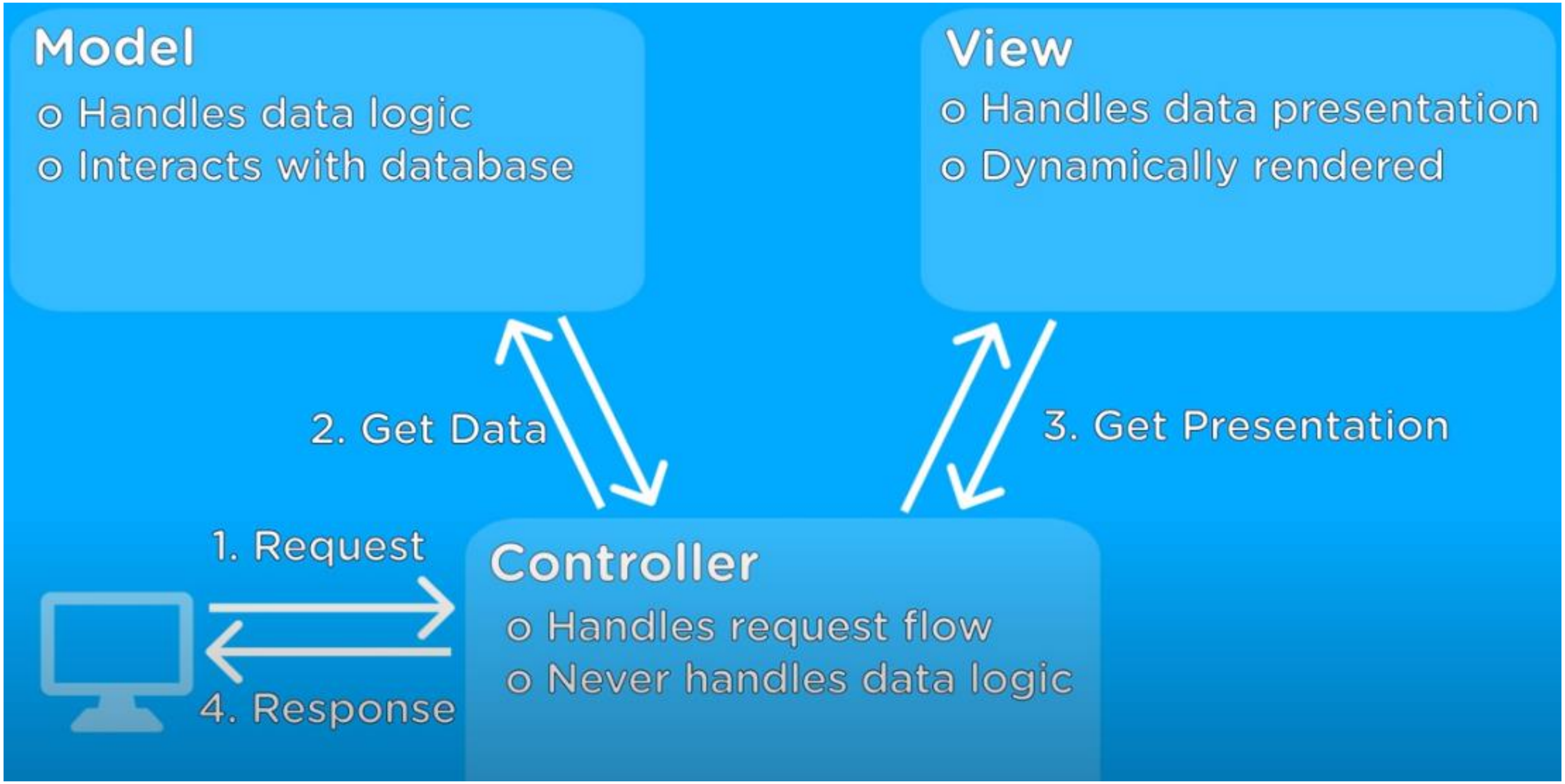
(birkan.yilmaz@bogazici.edu.tr)

Adapted from slides of Dr. Albert Ali Salah

MVC Architecture / Design Pattern

- Goal
  - Split the application into specific sections with their own purpose
- Controller
  - Handles request flow
  - Simple code
- Model
  - Handles data logic
- View
  - Handles data presentations

MVC Design Pattern

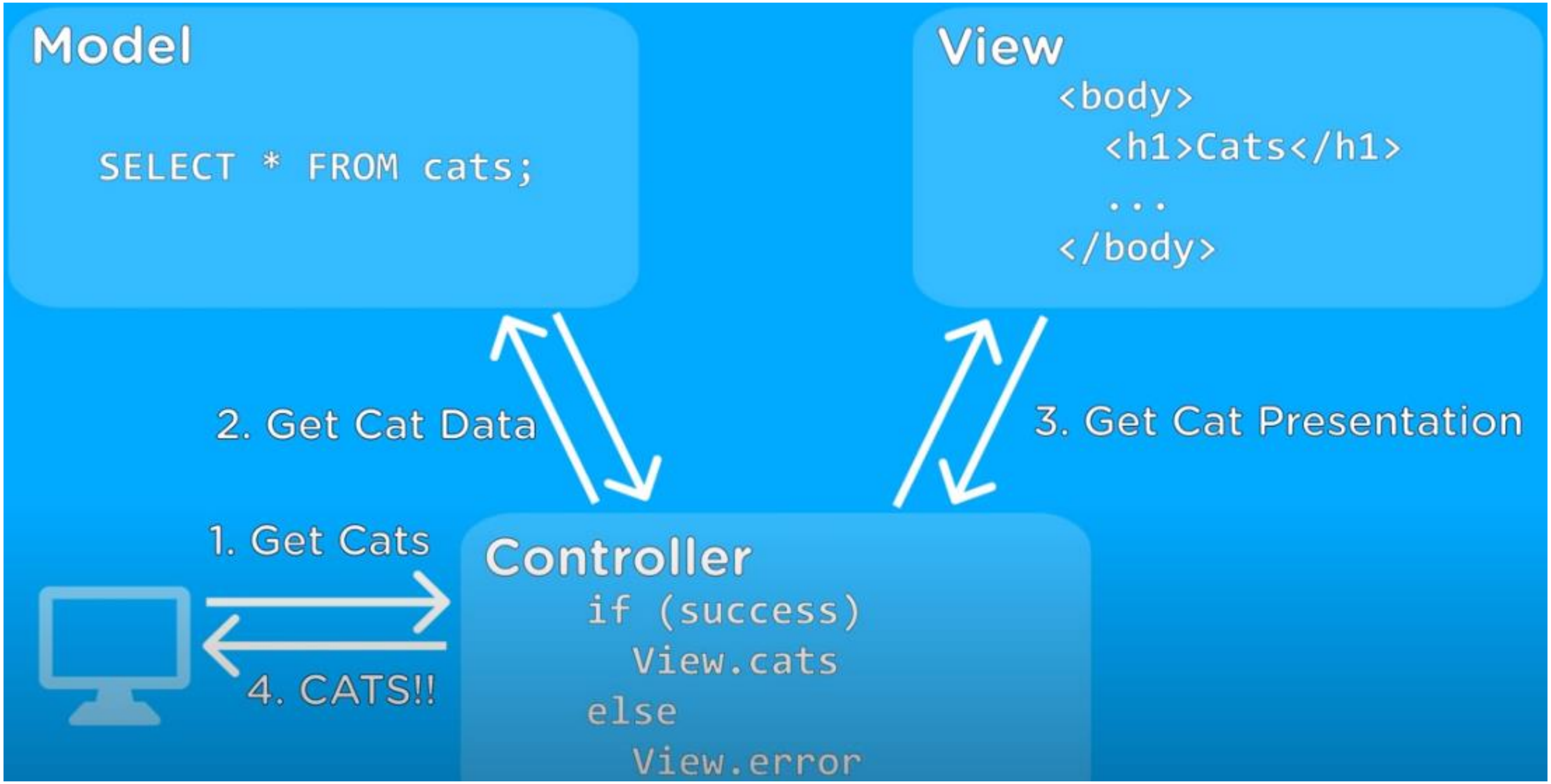


Model
- Handles data logic
- Interacts with database

View
- Handles data presentation
- Dynamically rendered

Controller
- Handles request flow
- Never handles data logic

MVC Example



```
SELECT * FROM cats;
```

```
<body>
  <h1>Cats</h1>
  ...
</body>
```

```
if (success)
  View.cats
else
  View.error
```

MVC Design Pattern

- View
  - What user sees, format template
  - Ex: Web pages CSS … etc.
- Model
  - Describes actual objects of the application and any code that is associated with them
  - Ex: Youtube video and channel, both are parts of model for the Youtube case
- Controller
  - Middleman that routes everything

MVC Pros & Cons

- Pros
  - Separation of concerns
  - Changing one part is isolated from others
  - Teams can be organized in a better way
- Cons
  - Increased number of files

Discussion

- What happens when you click on a Youtube video assuming that the MVC pattern is used at the background?

Blogging Website Example Model: Main Entities

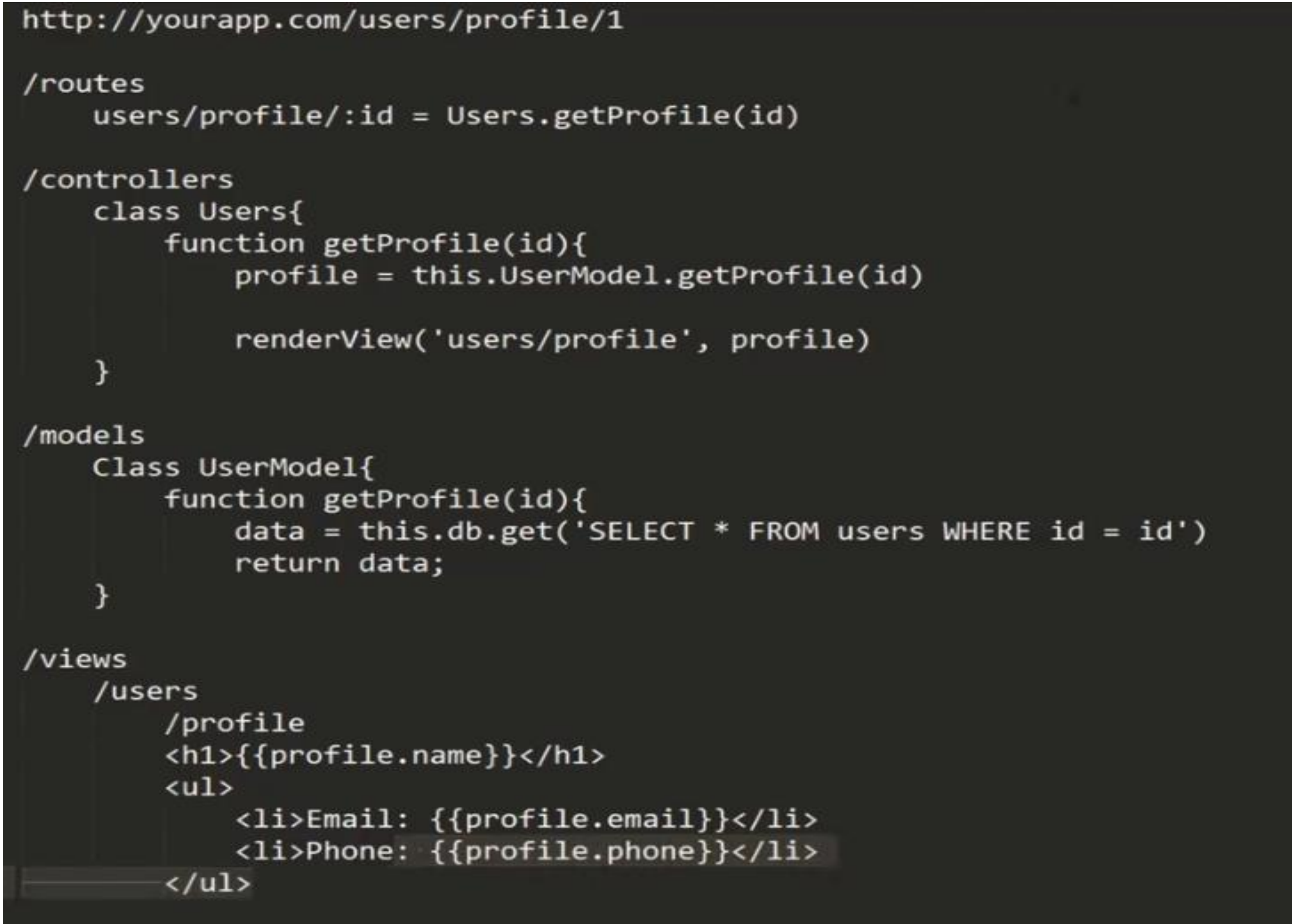
- Users
- Posts
- Comments

Blogging Website Example Model: Classes & Scripts

- Users
  - UserModel
  - UserController
  - UserView(s)
- Posts
  - PostModel
  - PostController
  - PostView(s)
- Comments
  - Same

- UsersView(s)
  - vwUserEdit
  - vwUserShow
  - vwUserSignIn
  - vwUserSignUp

Simple Example: Under MVC Folders



```
http://yourapp.com/users/profile/1

/routes
    users/profile/:id = Users.getProfile(id)

/controllers
    class Users{
        function getProfile(id){
            profile = this.UserModel.getProfile(id)

            renderView('users/profile', profile)
    }

/models
    Class UserModel{
        function getProfile(id){
            data = this.db.get('SELECT * FROM users WHERE id = id')
            return data;
    }

/views
    /users
        /profile
        <h1>{{profile.name}}</h1>
        <ul>
            <li>Email: {{profile.email}}</li>
            <li>Phone: {{profile.phone}}</li>
        </ul>
```

Activity: MVC Design Pattern

- You are required to design “An instant messaging application designed for teams with the following features.”
  - Joining to a workspace via email and link
  - Open and protected channels inside workspace
  - In channels there should be a task management tab for simple task management operations such as assigning tasks to users and viewing the status of the tasks
- First Step: Requirements Elicitation