CS 350 – Software Engineering

October 20, 2017

Week 9, Class #26

Data Design

Mark Seaman

MWF – 10:00-11:30 - Kepner 0095F

This Week – Design

- Monday, 10-16
  - Lecture – Architecture
- Wednesday, 10-18
  - Lecture – Detailed Design
- Friday, 10-20
  - Lecture – Data Design

Next Week

Code

Version Control Exercise

- ✧ Create a folder in the Exercise Results with your BearID
- ✧ Convert your plan into Markdown
  - Project Plan
  - Technology Plan
  - Design Plan

Design

✧ Architecture

- System partitioning into components

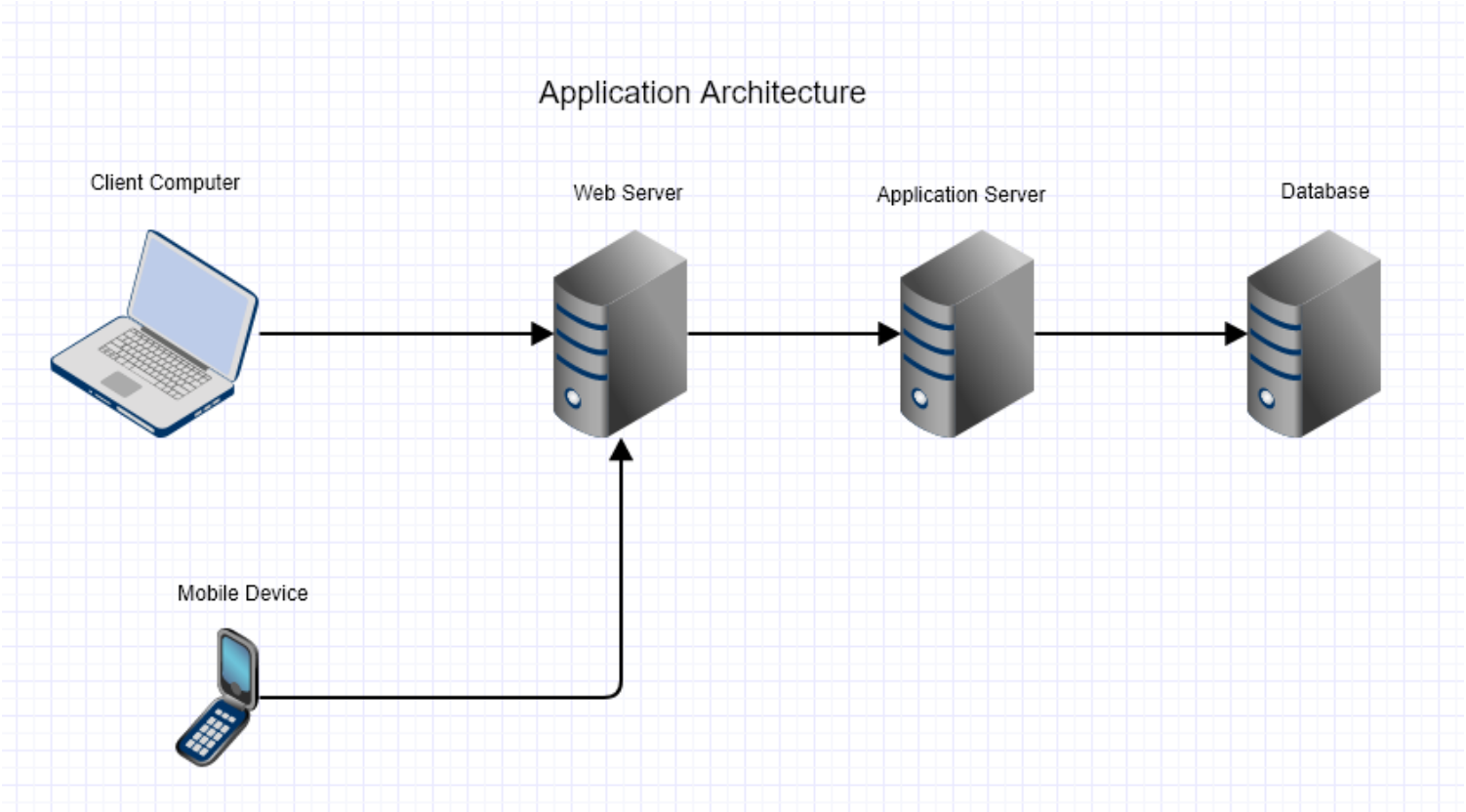
✧ Interface Design

- Describe each interface as a function stack

✧ Data Design

- Identify the key object classes
- Define the properties in each class
- Relationships between objects

App Architecture

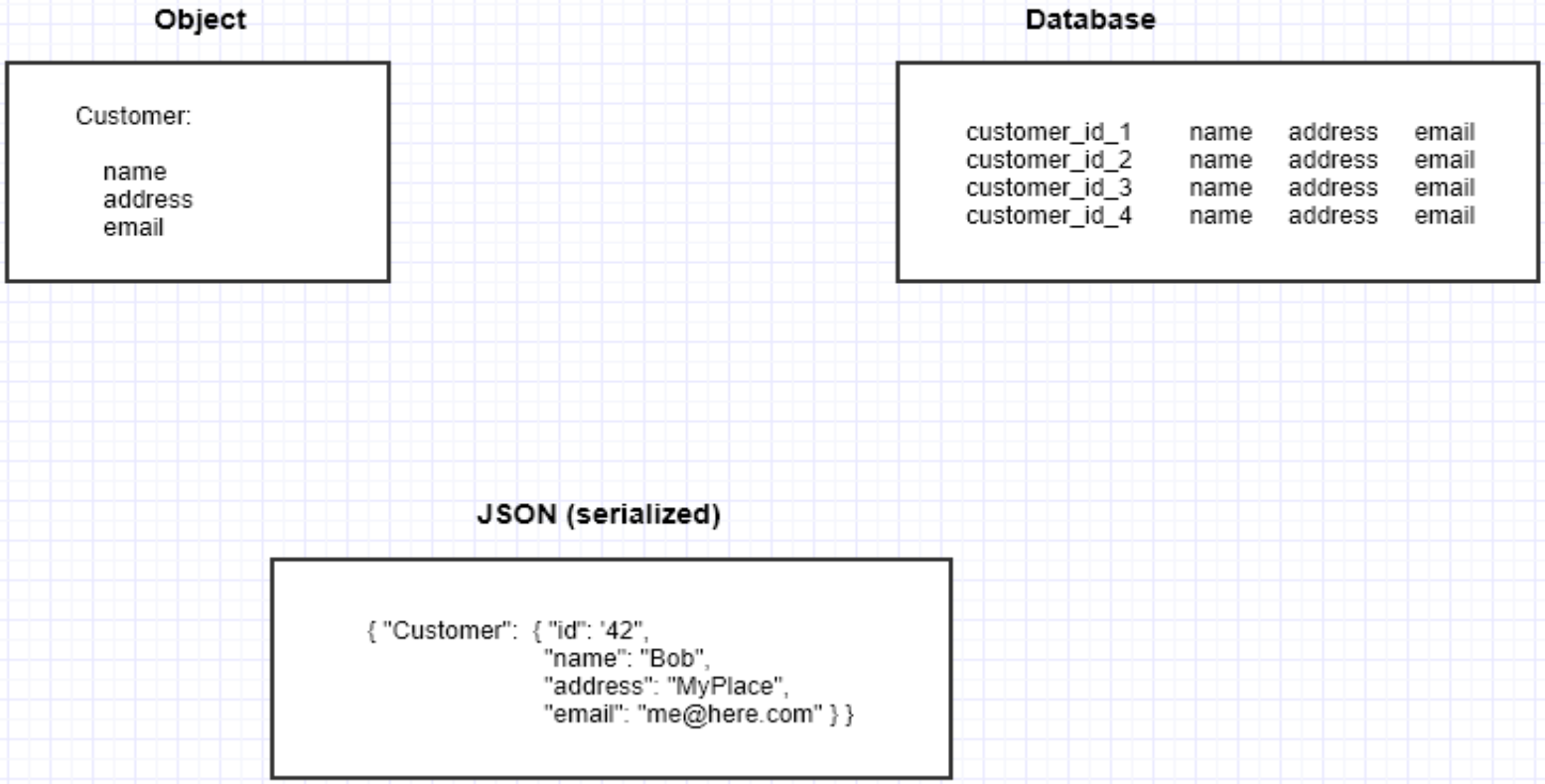


Object Relational Mapping - ORM

✧ Perspectives

- Database – rows and columns
- Objects – class instances
- JSON – serialized text

Object Relational Mapping - ORM



Choose your language

- SQL
  - String sql = "SELECT ... FROM persons WHERE id = 10";
  - DbCommand cmd = **new** DbCommand(connection, sql);
  - Result res = cmd.Execute();
  - String name = res[0]["FIRST_NAME"];

- JavaScript
  - Person p = Person.Get(10);
  - String name = p.getFirstName();

Popular Framework ORMs

- ✧ Microsoft LINQ
- ✧ Rails - Active Record
- ✧ Django ORM

Architecture Process

- ✧ Partition the system
- ✧ Define interfaces
- ✧ Create a test plan
- ✧ Manage dependencies

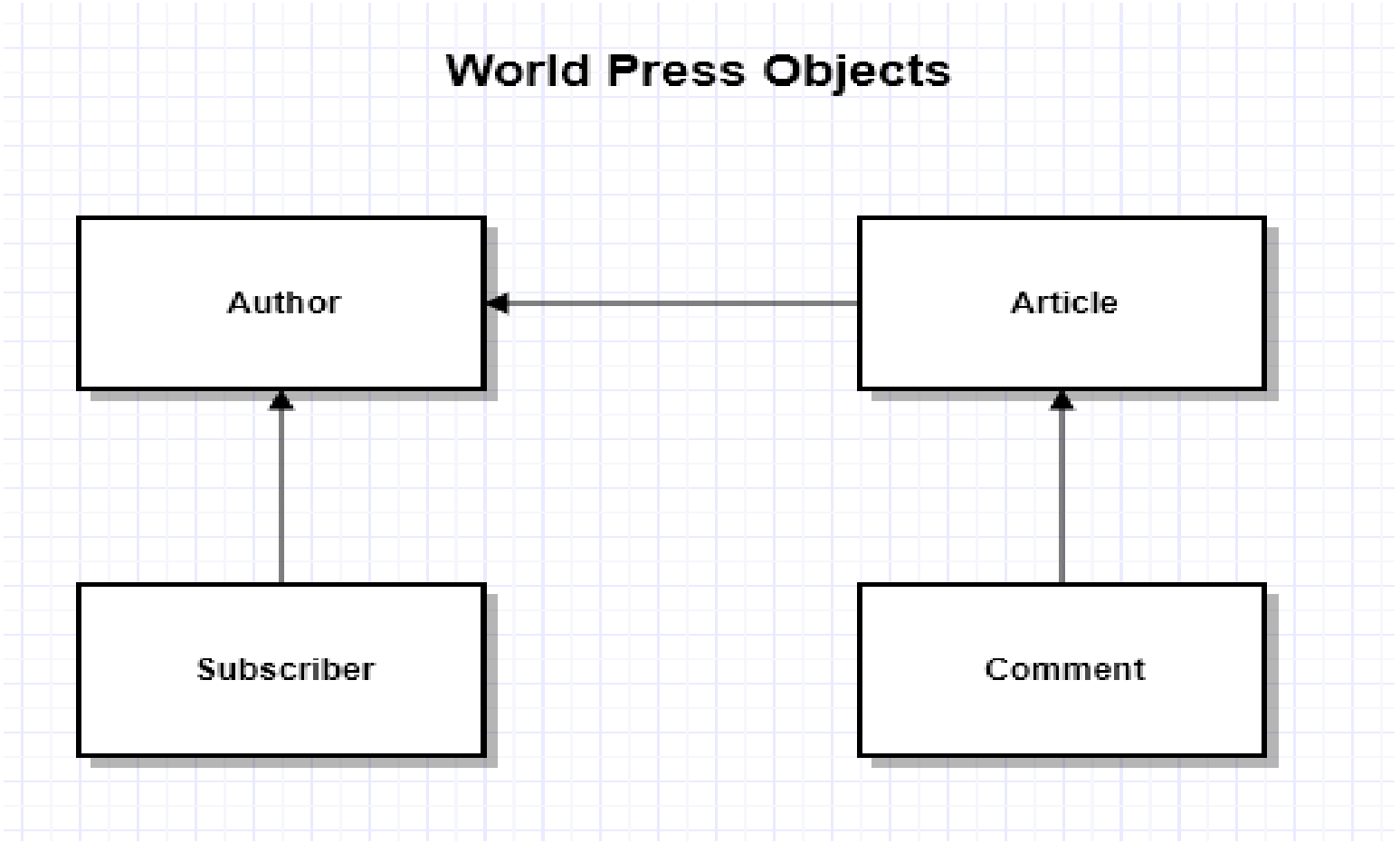
World Press Architecture

- Computer client - Web browser
- Web server - http, urls to web pages
- Email server - SMTP
- Application server – functions
- Database - CRUD

Data Design - CRUD

- Author
  - name, email, password
- Article
  - user, title, body
- Subscriber
  - email, user

World Press Data



World Press Classes

- class Author(models.Model):
  - name = models.CharField(max_length=100)
  - email = models.EmailField(max_length=100)
  - password = models.CharField(max_length=100)

- class Article(models.Model):
  - title = models.CharField(max_length=100)
  - body = models.TextField()
  - author = models.ForeignKey(Author)

World Press Classes


```
class Subscriber(models.Model):
    email = models.CharField(max_length=100)

class Comments(models.Model):
    text = models.CharField(max_length=1000)
    article = models.ForeignKey(Article)
```

Author CRUD

- Create
  - Author.objects.create(name='Bob', email='bob@here.com', password='12345')
- Read
  - Author.objects.all()
  - x = Author.objects.get(name='Bob')
- Update
  - x.email = 'Rachel'
  - x.save()
- Delete
  - x.delete()

Article CRUD

- Create
  - Article.objects.create(name=bob, title='No title', body=None)
- Read
  - Article.objects.all()
  - x = Author.objects.get(author.name='Bob')
- Update
  - x.title = 'Awesome article'
  - x.author = cindy
  - x.save()
- Delete
  - x.delete()

Upcoming Actions

- ✧ Markdown Exercise 10/20
- ✧ Design Plan 10/23
- ✧ Exam on Friday, 10/27

Web Server Interface – high-level

- Login
  - register (name, email, password)
  - login (name, password)
- Author
  - add_article (user, title, body)
  - edit_article (user, title, body)
  - delete_article (user, title)
- Reader
  - get_article (title)
  - list_article (user)

Web Server Interface – high-level

- ✧ News (ignore this for now)
  - register (email)
  - verify_email (token)
  - unsubscribe (email)

Power of Wishful Thinking

- ✧ Top-down design
- ✧ Bottom-up construction
- ✧ Middle-out testing

Web Server Interface - Functions

✧ Login

- def register (name, email, password):
  - return render("register.html", name, email, password)
- def login (name, password):
  - return render("login.html", name, email, password)

Web Server Interface - Functions

✧ Author

- def add_article (user, title, body):
  - return render("article_add.html", user, title, body)
- def edit_article (user, title, body):
  - return render("article_edit.html", user, title, body)
- def delete_article (user, title):
  - return render("article_delete.html", user, title)

Web Server Interface - Functions

✧ Reader

- def get_article (title):
  - return render("article_get.html", title)
- def list_article (user):
  - return render("article_list.html", user)