App Development & Modeling

BSc in Applied Computing

Produced by

Eamonn de Leastar (edeleastar@wit.ie)

Department of Computing, Maths & Physics
Waterford Institute of Technology

http://www.wit.ie

http://elearning.wit.ie

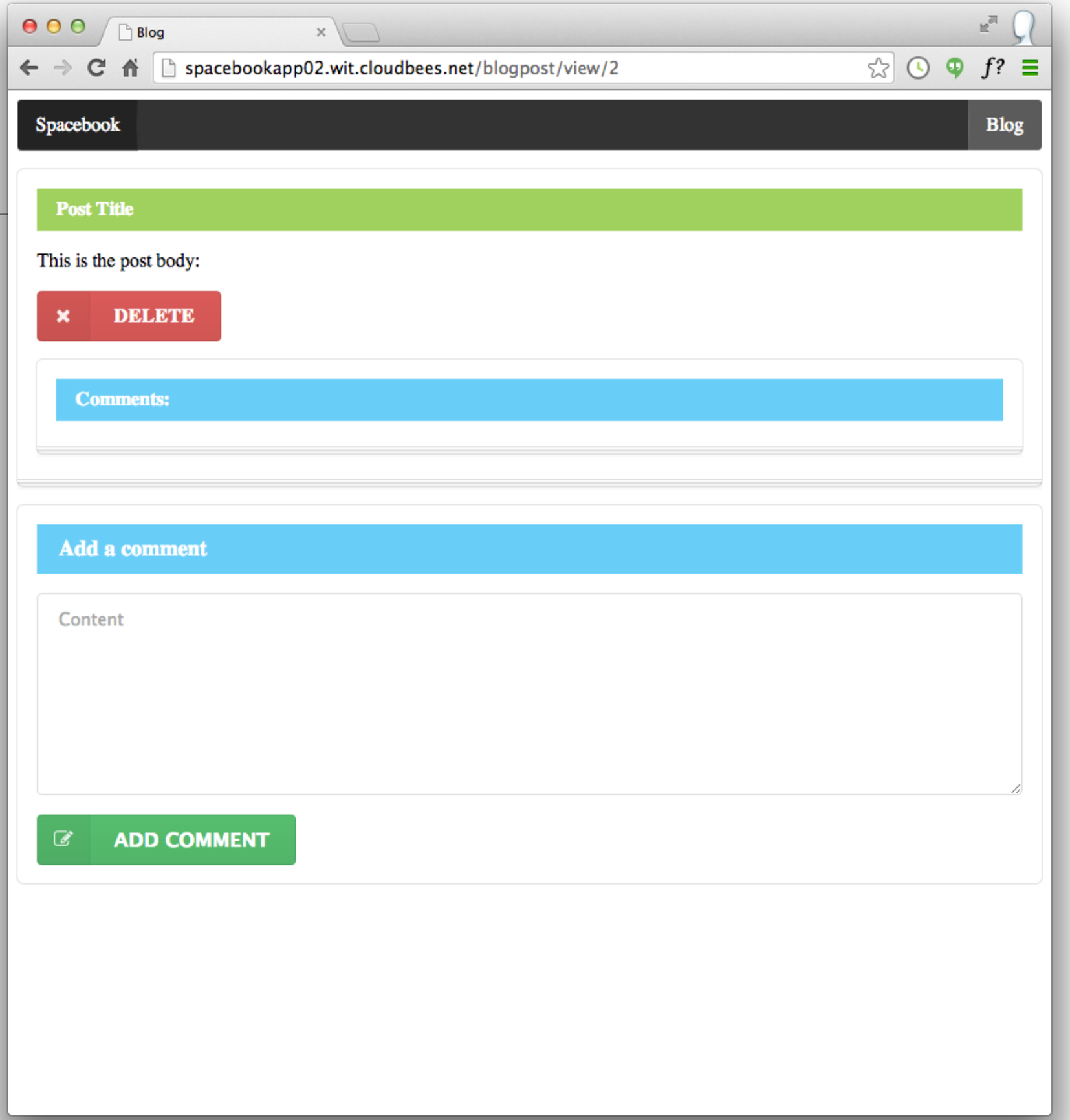
Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

eLearning support unit

Comment Feature

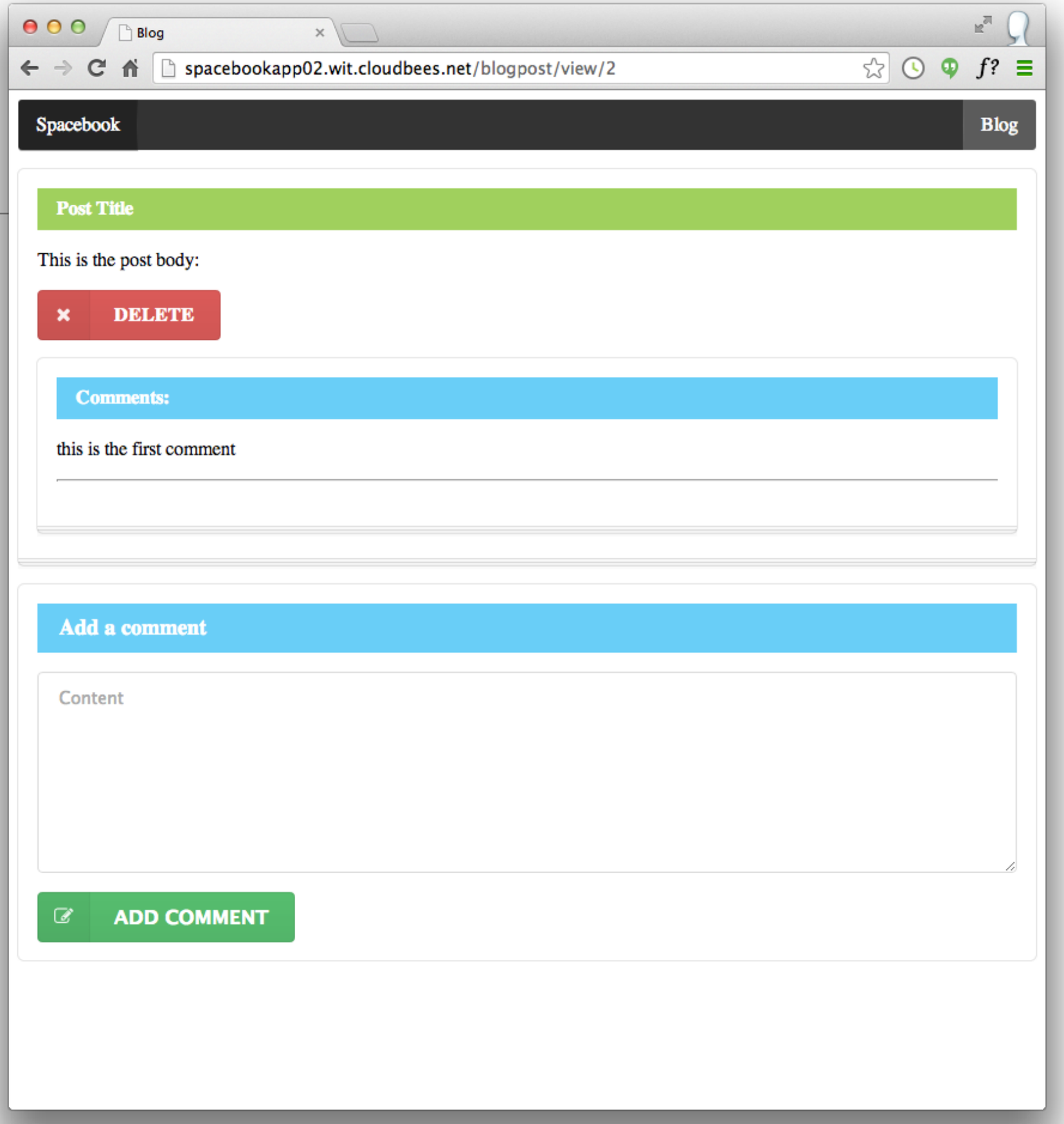
Post Comments

- New comments can be entered below the post



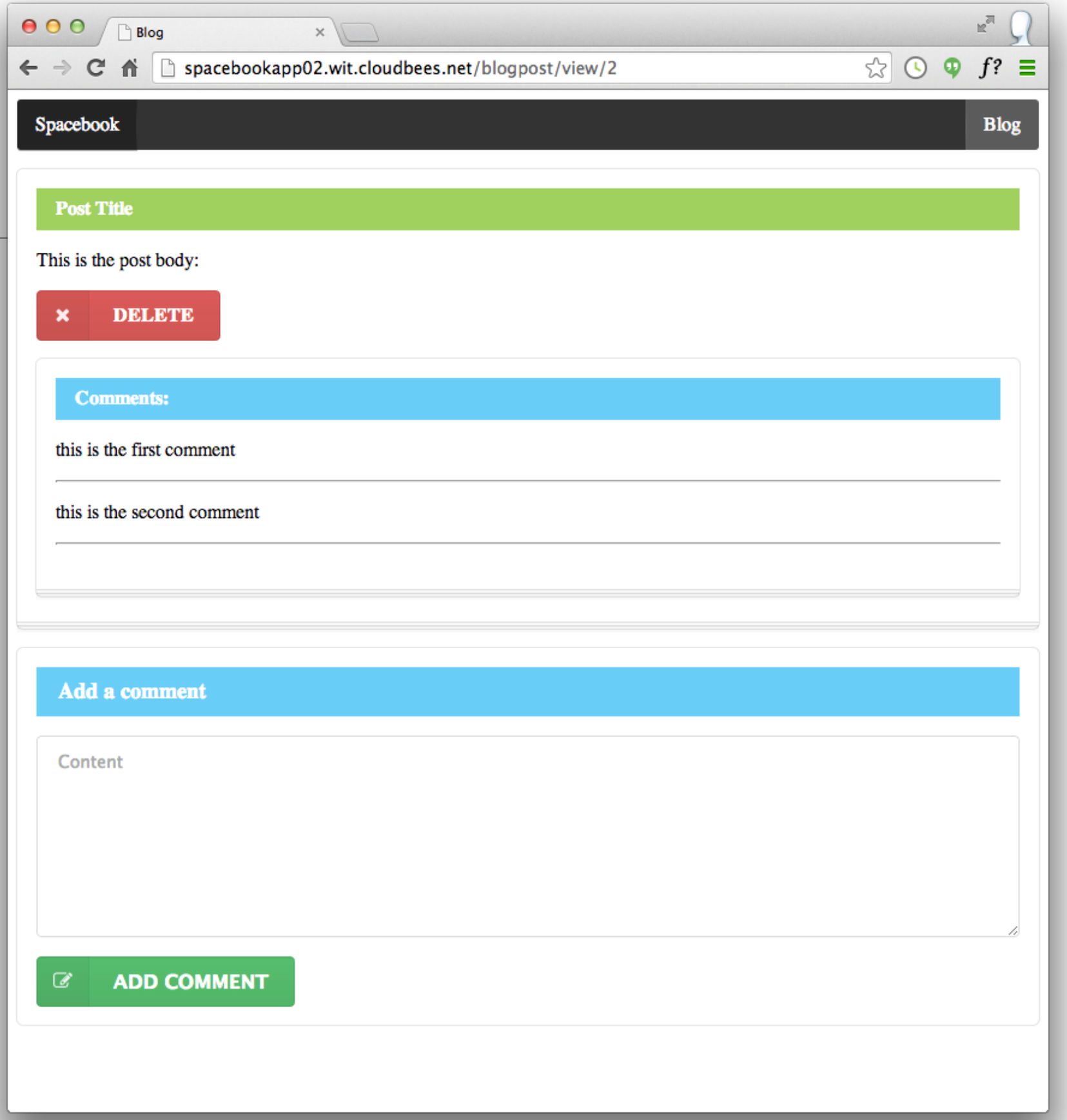
Post Comments

- List of comments appears under comment



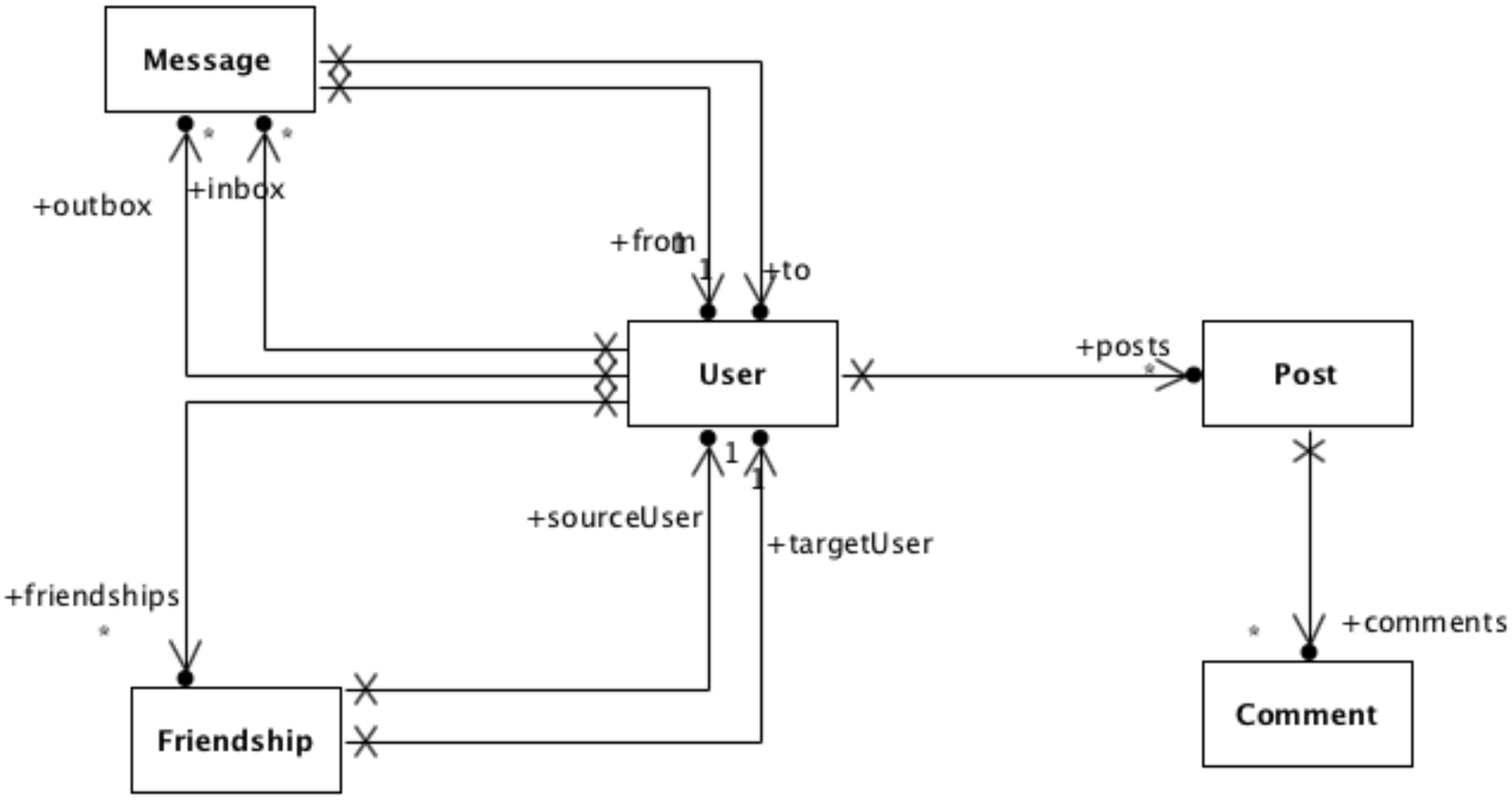
Post Comments

- Comments appear in order in which they are made

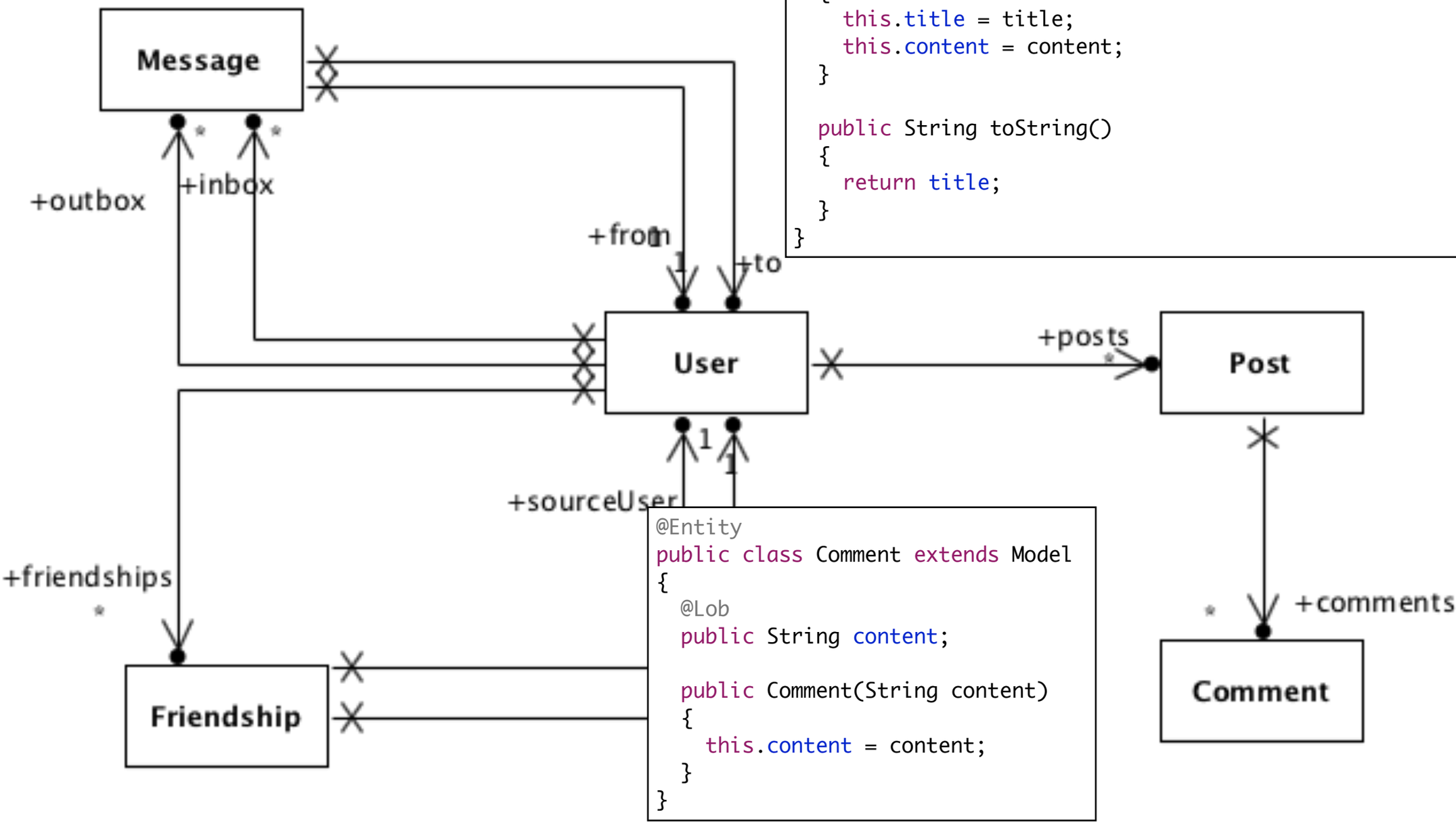


http://spacebookapp02.wit.cloudbees.net

Model First


Message
+outbox
+inbox
+from
1
+to
User
+posts
Post
1
1
+sourceUser
+targetUser
+friendships
*
+comments
*
Friendship
Comment

Model First




```
@Entity
public class Post extends Model
{
  public String title;
  @Lob
  public String content;

  @OneToMany  public List<Comment> comments = new ArrayList<Comment>();

  public Post(String title, String content)
  {
    this.title = title;
    this.content = content;
  }

  public String toString()
  {
    return title;
  }
}
```

```
@Entity
public class Comment extends Model
{
  @Lob
  public String content;

  public Comment(String content)
  {
    this.content = content;
  }
}
```

Post & Comment

- Posts maintain list of comments
- Write unit tests to:
  - Create comments
  - Delete comments

```
@Entity
public class Post extends Model
{
  public String title;
  @Lob
  public String content;

  @OneToMany  public List<Comment> comments = new ArrayList<Comment>();

  public Post(String title, String content)
  {
    this.title = title;
    this.content = content;
  }

  public String toString()
  {
    return title;
  }
}
```

```
@Entity
public class Comment extends Model
{
  @Lob
  public String content;

  public Comment(String content)
  {
    this.content = content;
  }
}
```

Comment Test Fixtures

```
public class CommentTest  extends UnitTest
{
  private User bob;
  private Post aPost;

  @BeforeClass
  public static void loadDB()
  {
    Fixtures.deleteAllModels();
  }

  @Before
  public void setup()
  {
    bob   = new User("bob", "jones", "bob@jones.com", "secret",20, "irish");
    aPost = new Post ("Post A", "This is the post A content");
    bob.posts.add(aPost);
    aPost.save();
    bob.save();
  }

  @After
  public void teardown()
  {
    bob.delete();
    aPost.delete();
  }
```

Comment Test (1)

```
@Test
public void testAddComment()
{
  Comment comment1 = new Comment("Comment 1");
  comment1.save();

  User user = User.findByEmail("bob@jones.com");
  Post post = user.posts.get(0);
  post.comments.add(comment1);
  post.save();
  user.save();

  User anotherUser = User.findByEmail("bob@jones.com");
  assertEquals("Comment 1", anotherUser.posts.get(0).comments.get(0).content);
}
```

Comment Test (2)

```
 @Test
public void testAddDeleteComment()
{
  Comment comment1 = new Comment("Comment 1");
  comment1.save();

  User user = User.findByEmail("bob@jones.com");
  Post post = user.posts.get(0);
  post.comments.add(comment1);
  post.save();
  user.save();

  User anotherUser = User.findByEmail("bob@jones.com");
  assertEquals("Comment 1", anotherUser.posts.get(0).comments.get(0).content);

  post.comments.clear();
  post.save();
  comment1.delete();

  User yetAnotherUser = User.findByEmail("bob@jones.com");
  assertEquals(0, yetAnotherUser.posts.get(0).comments.size());
}
```

BlogPost Controller

- New Action:
  - newComment - create a new comment

```
public class BlogPost  extends Controller
{
  public static void view(Long postid)
  {
    Logger.info("Post ID = " + postid);
    Post post = Post.findById(postid);
    render (post);
  }

  public static void newComment(Long postid, String content)
  {
       ?
  }
}
```

BlogPost/view.html