# *The Jungle Game*

**Chesshire Coders**

➢ Angélica Fallas
➢ Taner King
➢ Adam Gundem
➢ Alexander Hennings
➢ Cameron Ackerman

# *Logical Architecture*

## Domain

edu.colostate.cs.cs414.chesshireCoders.jungleClient.network

edu.colostate.cs.cs414.chesshireCoders.jungleUtil

edu.colostate.cs.cs414.chesshireCoders.jungleUtil.events

edu.colostate.cs.cs414.chesshireCoders.jungleUtil.game

edu.colostate.cs.cs414.chesshireCoders.jungleUtil.listeners

edu.colostate.cs.cs414.chesshireCoders.jungleUtil.requests

edu.colostate.cs.cs414.chesshireCoders.jungleUtil.responses

edu.colostate.cs.cs414.chesshireCoders.jungleUtil.security

edu.colostate.cs.cs414.chesshireCoders.jungleUtil.types

## Technical Services

edu.colostate.cs.cs414.chesshireCoders.jungleServer.service

edu.colostate.cs.cs414.chesshireCoders.jungleServer.service.impl

## UI

edu.colostate.cs.cs414.chesshireCoders.jungleClient.ui

## Application

edu.colostate.cs.cs414.chesshireCoders.jungleClient.app

edu.colostate.cs.cs414.chesshireCoders.jungleClient.account

edu.colostate.cs.cs414.chesshireCoders.jungleClient.game

edu.colostate.cs.cs414.chesshireCoders.jungleClient.client

edu.colostate.cs.cs414.chesshireCoders.jungleClient.handler
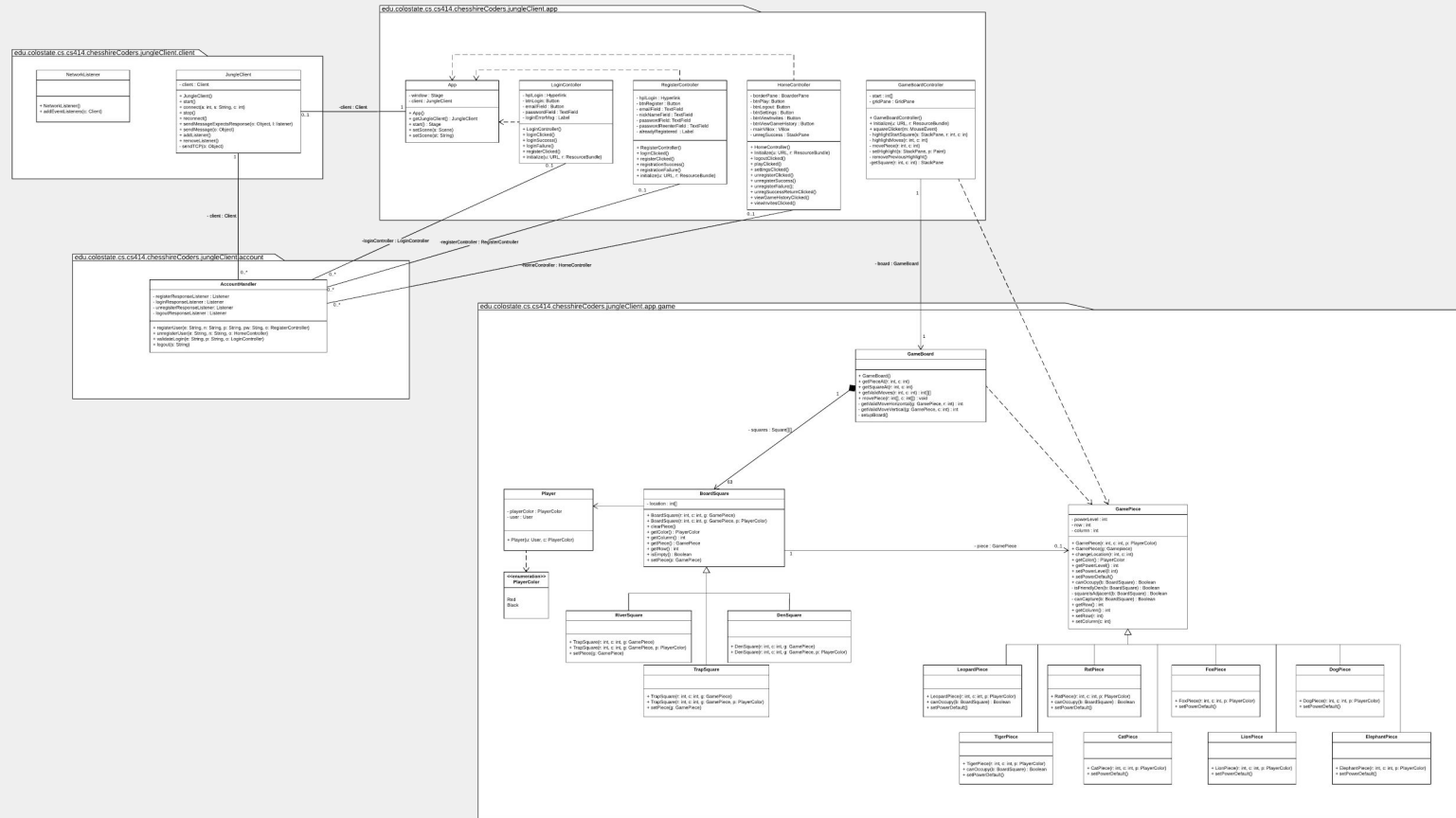
## Foundation

edu.colostate.cs.cs414.chesshireCoders.jungleServer.persistance

edu.colostate.cs.cs414.chesshireCoders.jungleServer.persistance.dao

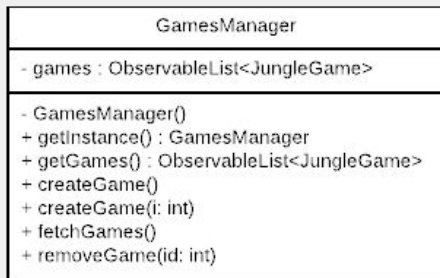edu.colostate.cs.cs414.chesshireCoders.jungleServer.dao.postgres
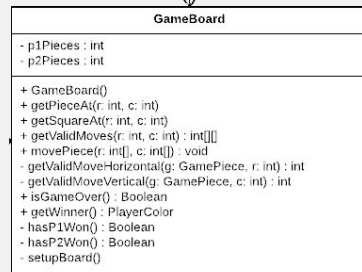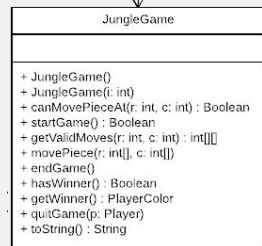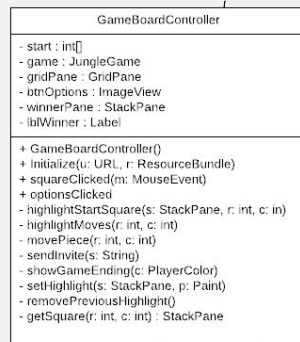
# Design Class Diagram: Jungle Client - Before

# Design Class Diagram: Jungle Client - After

# Design Class Diagram: Jungle Game

## Addition of Singleton Manager Classes

### AuthTokenManager

- authToken : AuthToken
- email : String

---

- AuthTokenManager()
- AuthTokenManager(a: AuthToken)
+ getToken() : AuthToken
+ getExpiration() : Date
+ renewExpirationDate(d: Date)
+ setAuthToken(t: AuthToken)
+ getEmail() : String
+ setEmail(e: String)
+ save(s: String)
+ load(s: String

### InviteManager

- invites : ObservableList<Invitation>

---

- InviteManager()
+ getInstance() : InviteManager
+ getInvites() : ObservableList<Invitation>
+ addInvitation(i: Invitation)
+ addInvitationEvent(e: InvitationEvent)
+ removeInvitation(i: Invitation)
+ acceptInvitation(i: Invitation)
+ rejectInvitation

### GamesManager

- games : ObservableList<JungleGame>

---

- GamesManager()
+ getInstance() : GamesManager
+ getGames() : ObservableList<JungleGame>
+ createGame()
+ createGame(i: int)
+ fetchGames()
+ removeGame(id: int)

## Addition of JungleGame Class

### GameBoardController

- start : int[]
- gridPane : GridPane

---

+ GameBoardController()
+ Initialize(u: URL, r: ResourceBundle)
+ squareClicker(m: MouseEvent)
- highlightStartSquare(s: StackPane, r: int, c: in)
- highlightMoves(r: int, c: int)
- movePiece(r: int, c: int)
- setHighlight(s: StackPane, p: Paint)
- removePreviousHighlight()
-getSquare(r: int, c: int) : StackPane

### GameBoard

---

+ GameBoard()
+ getPieceAt(r: int, c: int)
+ getSquareAt(r: int, c: int)
+ getValidMoves(r: int, c: int) : int[][]
+ movePiece(r: int[], c: int[]) : void
- getValidMoveHorizontal(g: GamePiece, r: int) : int
- getValidMoveVertical(g: GamePiece, c: int) : int
- setupBoard()

### GameBoardController

- start : int[]
- game : JungleGame
- gridPane : GridPane
- btnOptions : ImageView
- winnerPane : StackPane
- lblWinner : Label

---

+ GameBoardController()
+ Initialize(u: URL, r: ResourceBundle)
+ squareClicked(m: MouseEvent)
+ optionsClicked
- highlightStartSquare(s: StackPane, r: int, c: in)
- highlightMoves(r: int, c: int)
- movePiece(r: int, c: int)
- sendInvite(s: String)
- showGameEnding(c: PlayerColor)
- setHighlight(s: StackPane, p: Paint)
- removePreviousHighlight()
- getSquare(r: int, c: int) : StackPane

### JungleGame

---

+ JungleGame()
+ JungleGame(i: int)
+ canMovePieceAt(r: int, c: int) : Boolean
+ startGame() : Boolean
+ getValidMoves(r: int, c: int) : int[][]
+ movePiece(r: int[], c: int[])
+ endGame()
+ hasWinner() : Boolean
+ getWinner() : PlayerColor
+ quitGame(p: Player)
+ toString() : String

### GameBoard

- p1Pieces : int
- p2Pieces : int

---

+ GameBoard()
+ getPieceAt(r: int, c: int)
+ getSquareAt(r: int, c: int)
+ getValidMoves(r: int, c: int) : int[][]
+ movePiece(r: int[], c: int[]) : void
- getValidMoveHorizontal(g: GamePiece, r: int) : int
- getValidMoveVertical(g: GamePiece, c: int) : int
+ isGameOver() : Boolean
+ getWinner() : PlayerColor
- hasP1Won() : Boolean
- hasP2Won() : Boolean
- setupBoard()
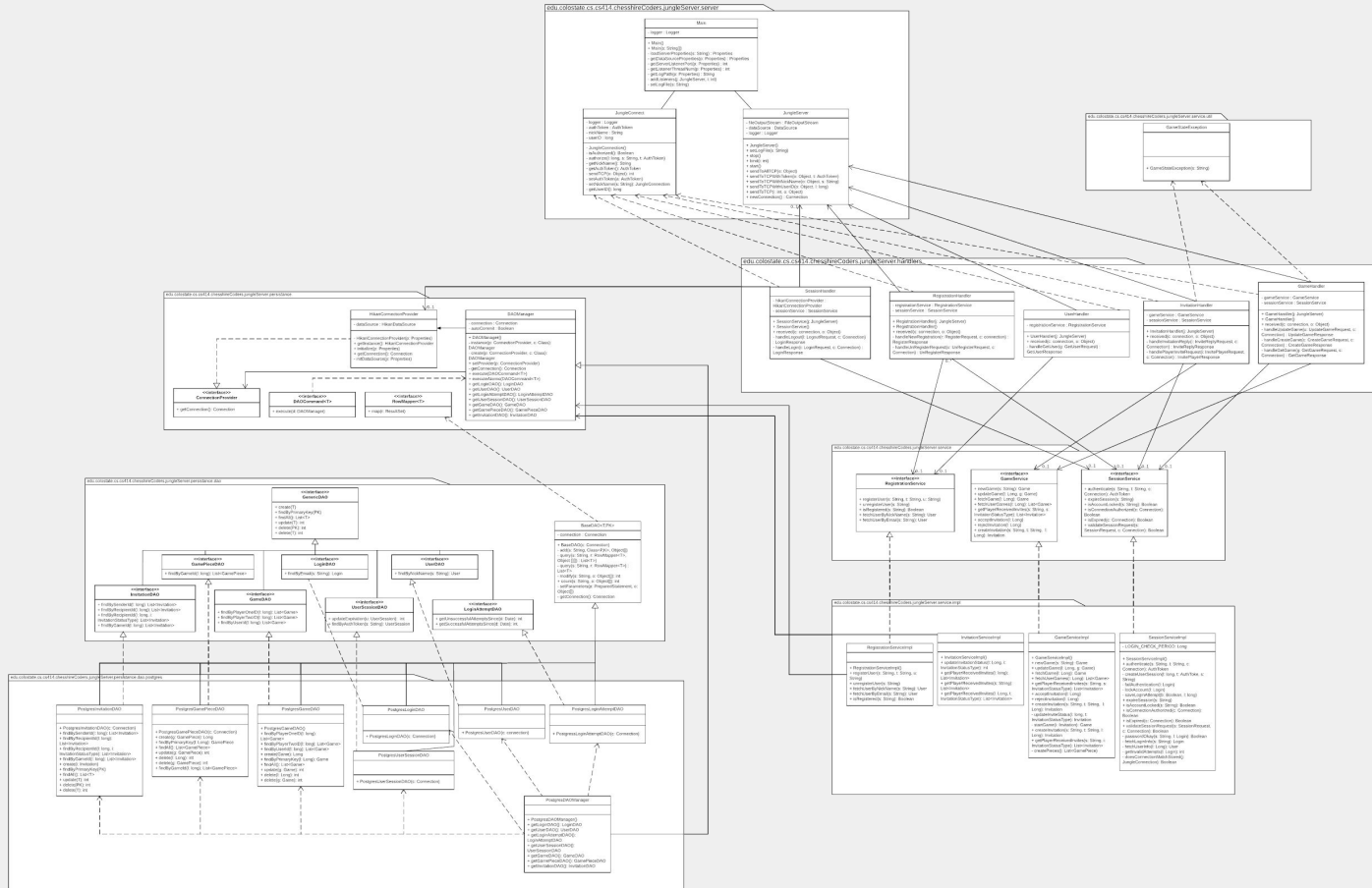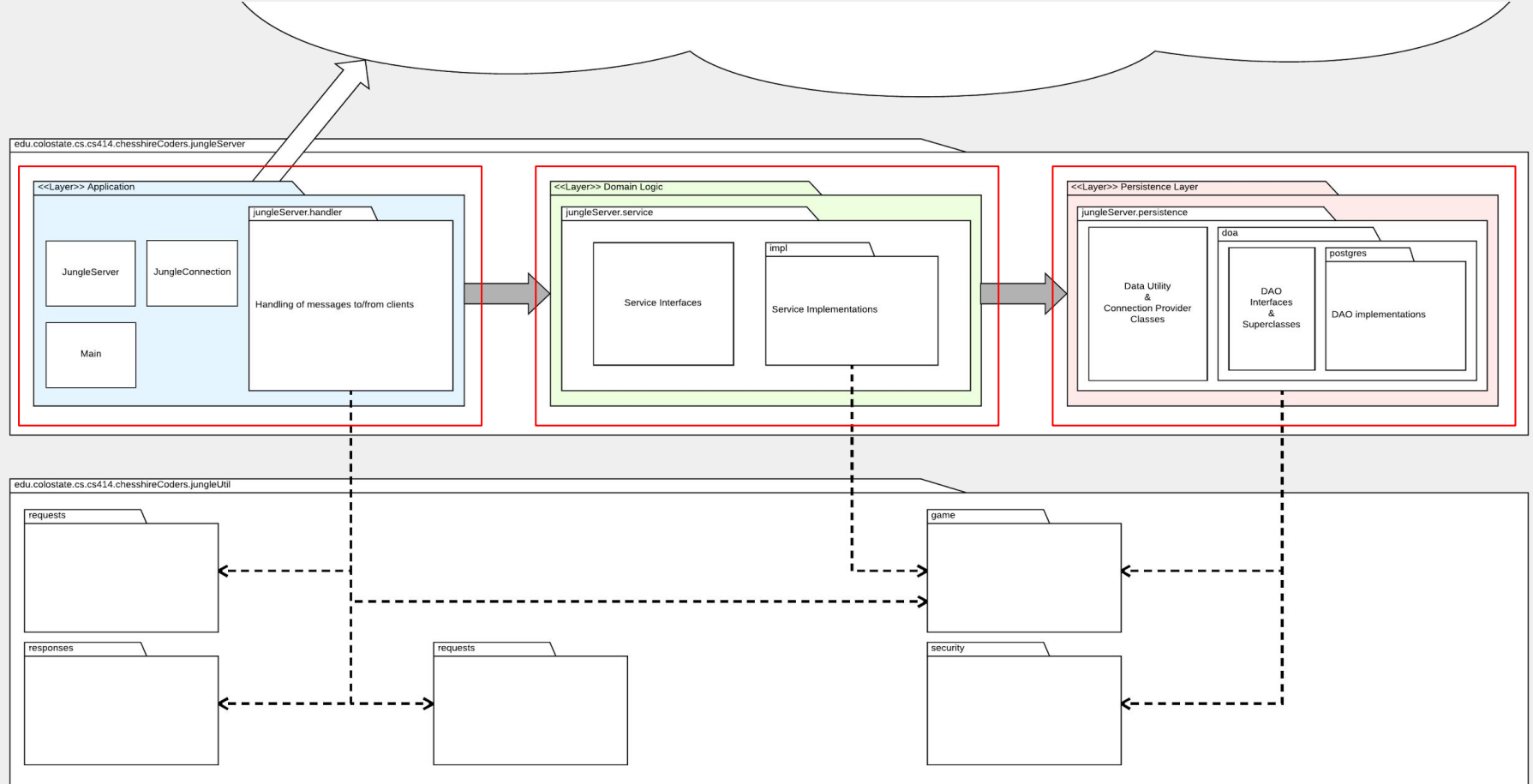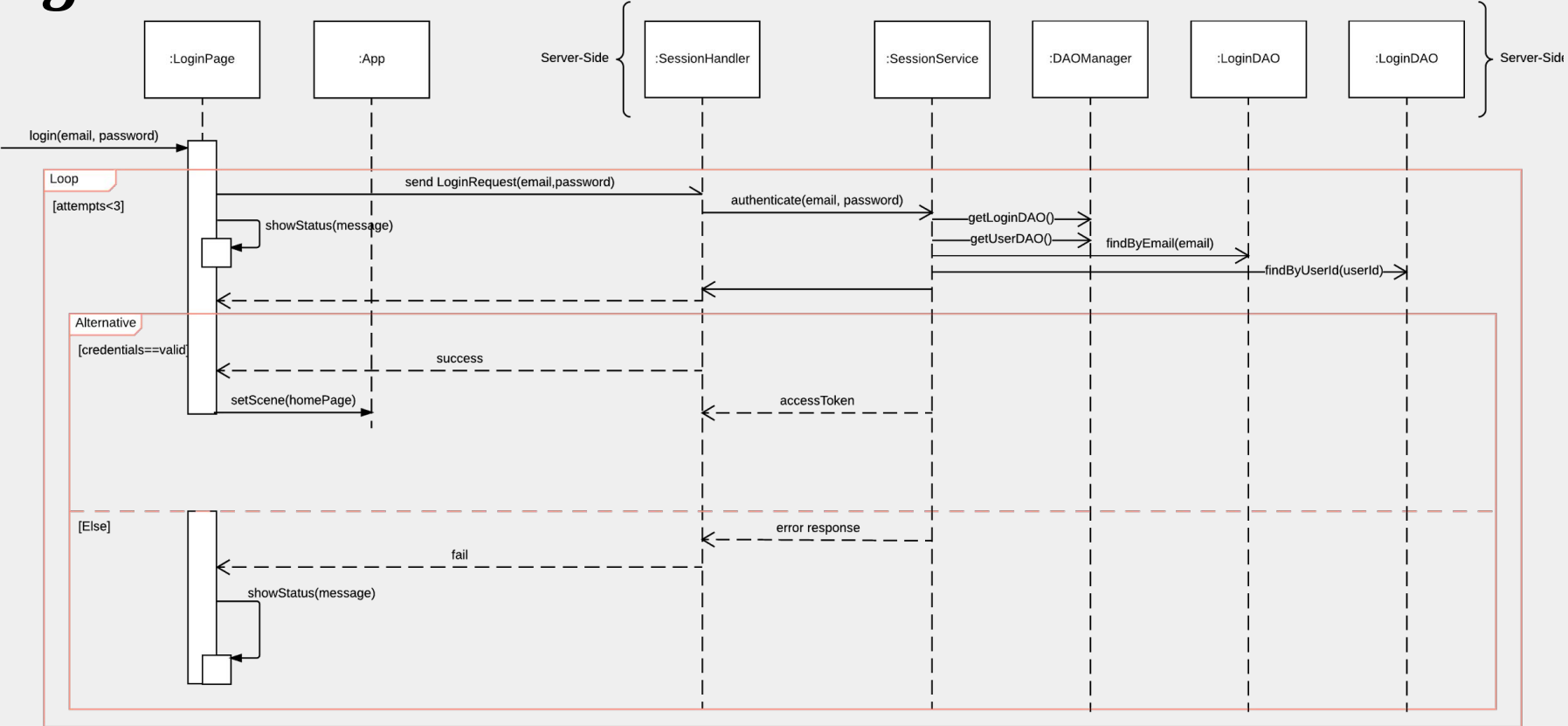
# Design Class Diagram: Jungle Network - Before

# Design Class Diagram: Jungle Server - After

# *Server Structure & Packages*



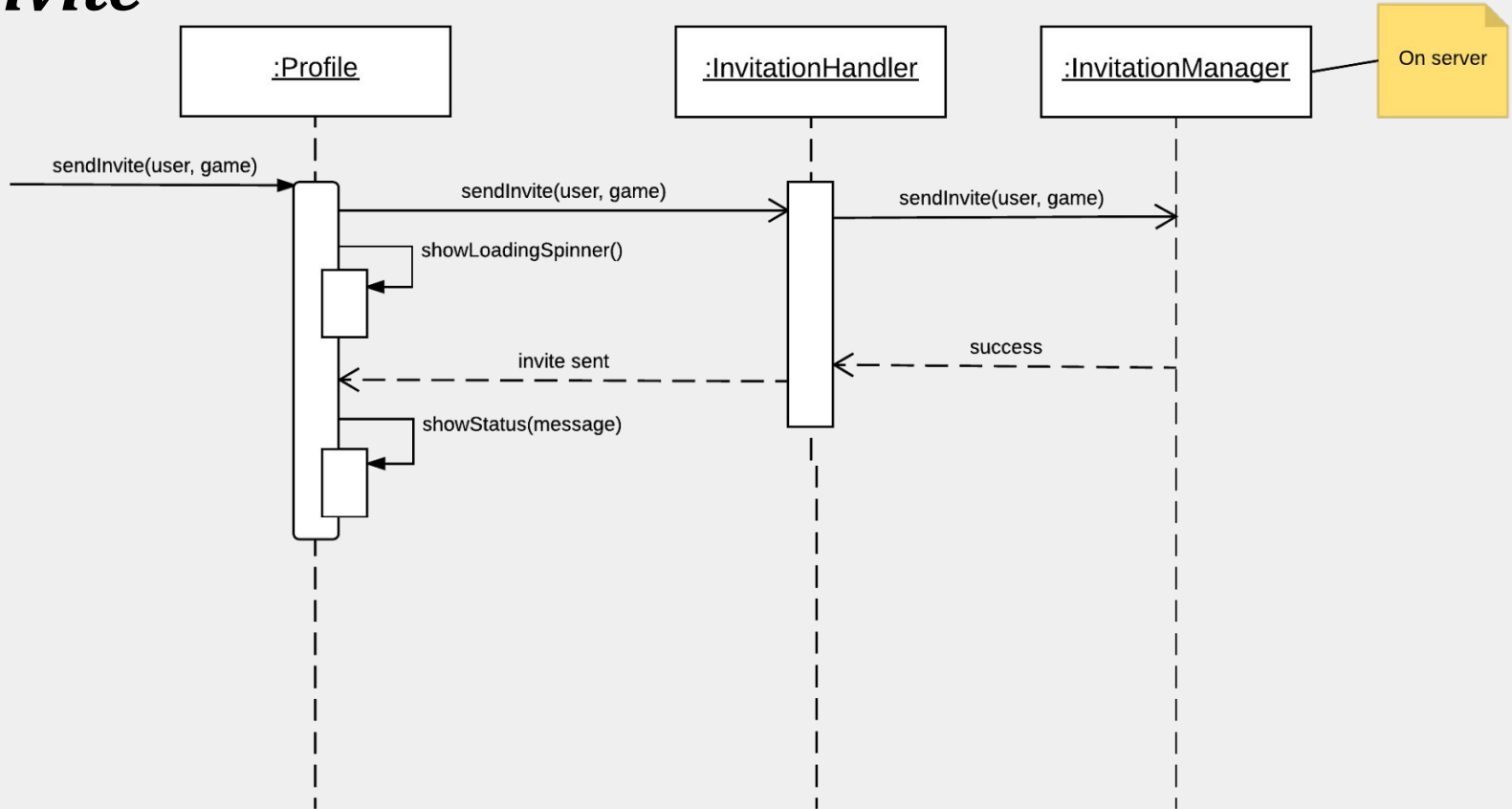**edu.colostate.cs.cs414.chesshireCoders.jungleServer**

**<<Layer>> Application**

jungleServer.handler

JungleServer

JungleConnection

Handling of messages to/from clients

Main

**<<Layer>> Domain Logic**

jungleServer.service

impl

Service Interfaces

Service Implementations

**<<Layer>> Persistence Layer**

jungleServer.persistence

doa

postgres

Data Utility
&
Connection Provider
Classes

DAO
Interfaces
&
Superclasses

DAO implementations

**edu.colostate.cs.cs414.chesshireCoders.jungleUtil**

requests

game

responses

requests

security

# Sequence Diagram: Login



Server-Side

:LoginPage  :App  :SessionHandler  :SessionService  :DAOManager  :LoginDAO  :LoginDAO  Server-Side

login(email, password)

**Loop**
[attempts<3]

send LoginRequest(email,password)

authenticate(email, password)

showStatus(message)

getLoginDAO()

getUserDAO()

findByEmail(email)

findByUserId(userId)

**Alternative**
[credentials==valid]

success

setScene(homePage)

accessToken

[Else]

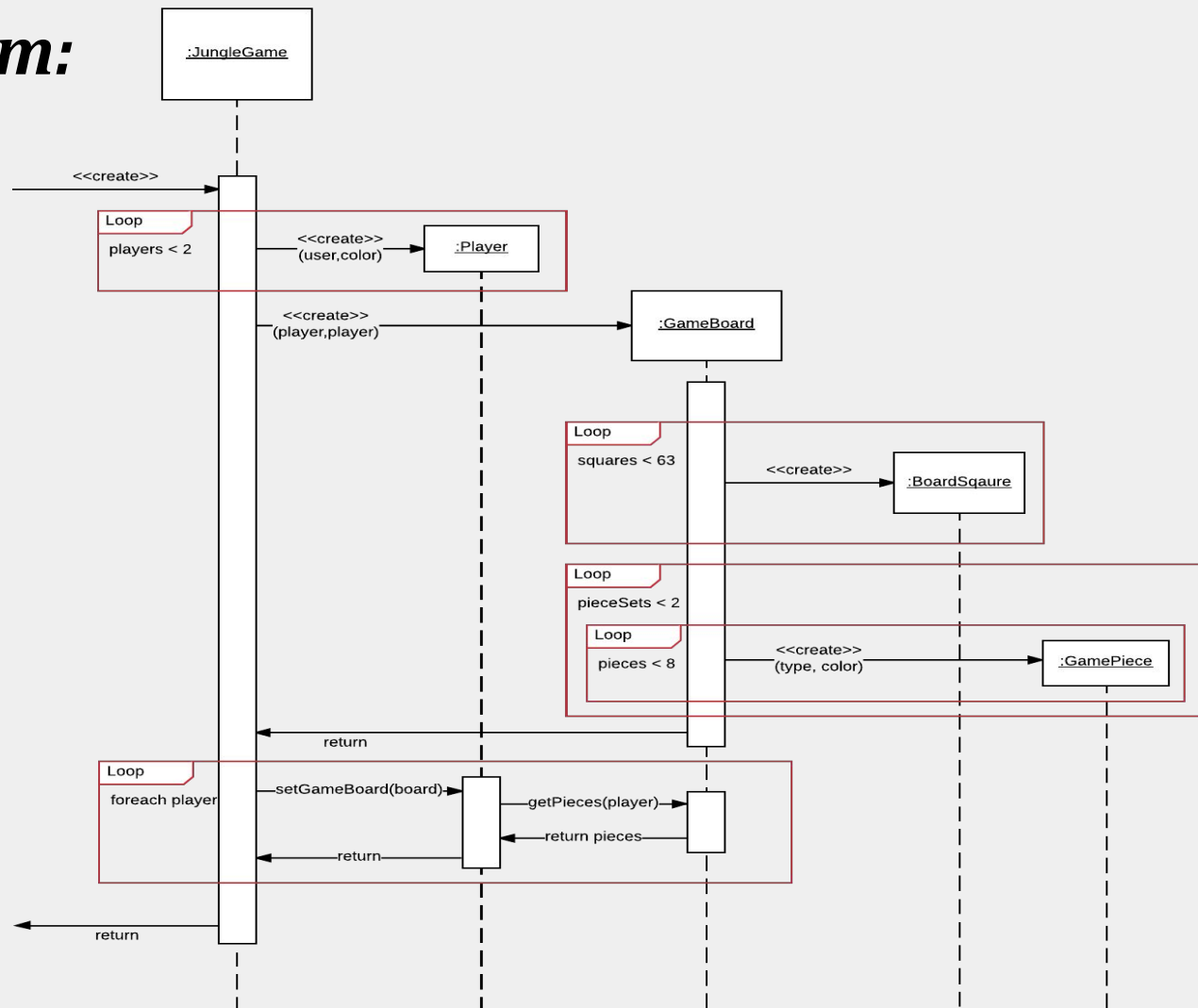error response

fail

showStatus(message)
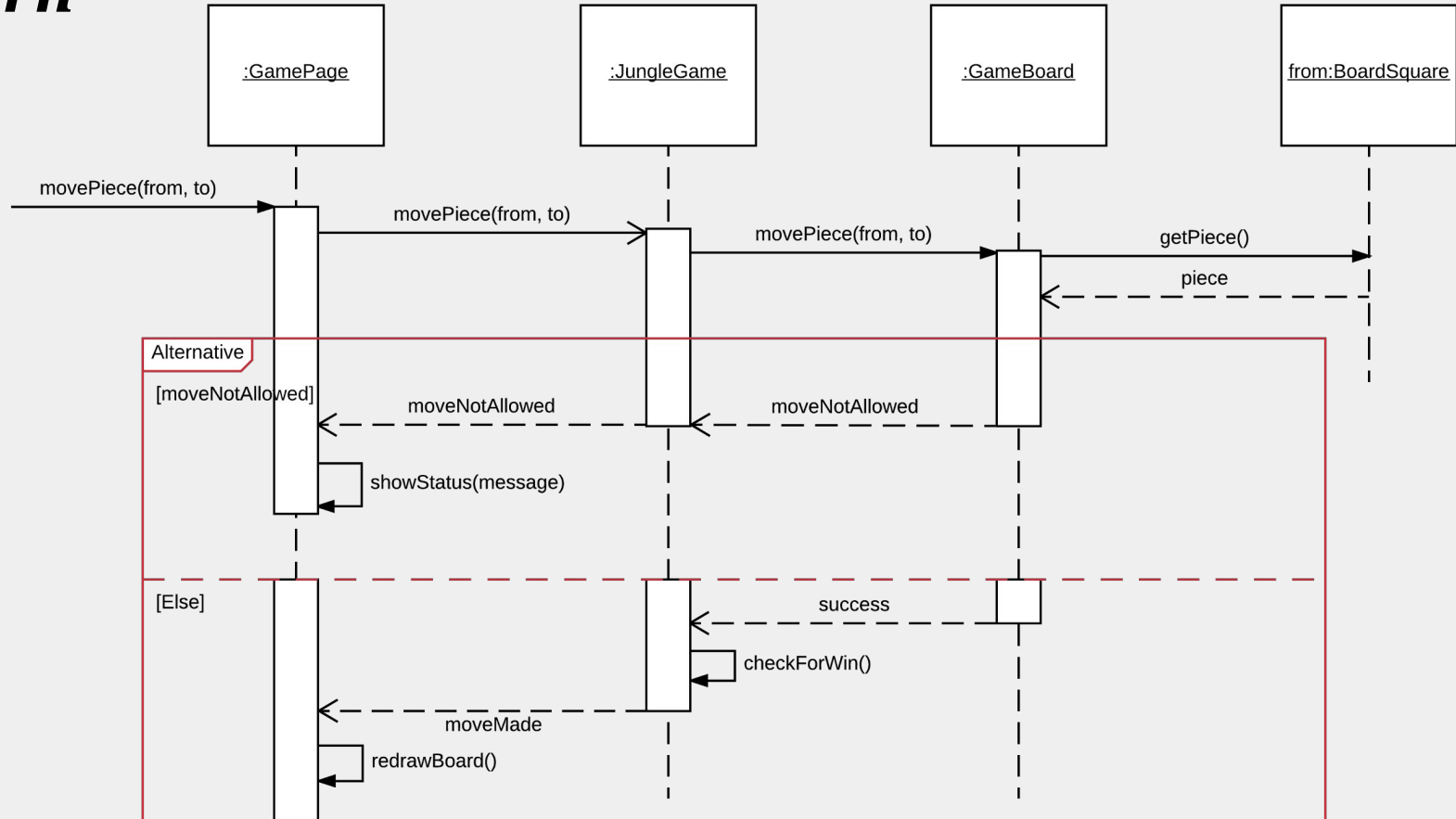
# Sequence Diagram: Register

# *Sequence Diagram: Game Invite*

# Sequence Diagram: Create Game

# Sequence Diagram: Take Turn

# *Refactoring and Design Pattern List*

- ➢ Singleton
- ➢ Factory Method
- ➢ Abstract Factory Method
- ➢ Prototype
- ➢ Facade
- ➢ Decorator

# Traceability Link Matrix

| | Login | Jungle Game | Game Board | Game Piece | Board Square | Player | User | Invite Manager | Invitation |
|---|---|---|---|---|---|---|---|---|---|
| [#1: Register to the system](#) | x | | | | | | x | | x |
| [#2: Create a new game](#) | x | x | x | | | | | | x |
| [#3: Invite other users to a game](#) | x | | | | | | | x | x |
| [#4: Respond to Game Invitation](#) | x | | x | | | x | | x | x |
| [#5: Quit Game](#) | x | x | x | | | x | | | |
| [#6: Unregister from System](#) | x | | | | | | x | | |
| [#7: View Player Profile](#) | x | | | | | x | | | |
| [#8: Log in to System](#) | x | | | | | | x | | |
| [#9: Log out of System](#) | x | | | | | | x | | |
| [#10: Move Game Piece](#) | x | x | x | x | x | x | | | |
| [#11: Switch Game](#) | x | x | x | | | x | | | |

*Demo*

# Questions
# And
# Discussion