***Progetto***

Movier

**INDICE**

1. Specifiche informali 1

2. Analisi e specifica dei requisiti 2

2.1 Analisi nomi-verbi 2

2.2 Revisione dei requisiti 2

2.3 Glossario dei termini 3

2.4 Classificazione dei requisiti 3

2.4.1 Requisiti funzionali 4

2.4.2 Requisiti sui dati 4

2.4.3 Vincoli / Altri requisiti 4

2.5 Modellazione dei casi d’uso 5

2.5.1 Attori e casi d’uso 5

2.5.2 Diagramma dei casi d’uso 5

2.5.3 Scenari 5

2.6 Diagramma delle classi 2

2.7 Diagrammi di sequenza 2

3. Stima dei costi 3

4. Piano di test funzionale 4

5. Progettazione 5

5.1 Diagramma delle classi 5

5.2 Diagrammi di sequenza 5

6. Implementazione 6

7. Testing 7

7.1 Test strutturale 7

7.1.1 Complessità ciclomatica 7

7.2 Test funzionale 8

# Specifiche informali

Si vuole realizzare un’applicazione che consiglia ad un utente quale film vedere.

Il sistema mostra all’utente un film e l’utente può indicarlo come like or not like.

L’interazione si ripete un numero definito di volte e poi si riavvia.

Il sistema permette all’utente di effettuare una ricerca by keyword sui titoli del film.

Un film è caratterizzato da un titolo, una descrizione, da una durata, da un anno di rilascio, da un rating, da un voto imdb, da una serie di piattaforme su cui è disponibile(anche nessuna).

Un film deve poter essere aggiunto o alla watchlist o alla lista “guarda più tardi”

I consigli automatizzati devono poter essere targhettizzati in base ad una o più piattaforme di streaming ed in base al fatto di essere da soli, con l’interesse amoroso, con gli amici e con la famiglia. Il sistema deve essere in grado di mostrare la watchlist e la lista “guarda più tardi”.

# Analisi e specifica dei requisiti

## Analisi nomi-verbi

Si vuole realizzare un’applicazione che consiglia ad un utente quale film vedere.

Il sistema mostra all’utente un film e l’utente può indicarlo come like or not like.

L’interazione si ripete un numero definito di volte e poi si riavvia.

Il sistema permette all’utente di effettuare una ricerca by keyword sui titoli del film.

Un film è caratterizzato da un titolo, una descrizione, da una durata, da un anno di rilascio, da un rating, da un voto imdb, da una serie di piattaforme su cui è disponibile(anche nessuna).

Un film deve poter essere aggiunto o alla watchlist o alla lista “watchLater”

I consigli automatizzati devono poter essere targhettizzati in base ad una o più piattaforme di streaming ed in base al fatto di essere da soli, con l’interesse amoroso, con gli amici e con la famiglia. Il sistema deve essere in grado di mostrare la watchlist e la lista “guarda più tardi”.

*Classi*

*Attori*

*Requisiti*

## Revisione dei requisiti

# N

# Il sistema deve mostrare all’utente un insieme di n film tenendo conto del fatto che l’utente possa essere da solo, in compagnia della fidanzata, della famiglia o degli amici. Se l’utente ha selezionato una o più piattaforme di streaming i consigli devono appartenere a quelle piattaforme

# Il sistema deve permettere all’utente di esprimere un giudizio binario su ciascun film

# Il sistema permette all’utente di ricercare tramite keyword un film sul titolo del film indipendentemente dalla piattaforma di streaming

# Il sistema deve permettere all’utente di aggiungere un film alla watchlist

# Il sistema deve permettere all’utente di aggiungere un film alla lista watchLater

# Il sistema deve permettere all’utente di visualizzare la watchList

# Il sistema deve permettere all’utente di visualizzare la watchLater

# Un film è caratterizzato da un titolo, da una descrizione,da una durata, da un anno di rilascio,da un rating, da un voto imdb e da una serie di piattaforme su cui è disponibile(anche nessuna)

## Classificazione dei requisiti

## Requisiti funzionali

|  |  |  |
| --- | --- | --- |
| ID | Requisito | Origine (n. frase dei requisiti revisionati) |
| RF01 | Il sistema deve mostrare all’utente un insieme di n film tenendo conto del fatto che l’utente possa essere da solo, in compagnia della fidanzata, della famiglia o degli amici. Se l’utente ha selezionato una o più piattaforme di streaming i consigli devono appartenere a quelle piattaforme | 1 |
| RF02 | Il sistema deve permettere all’utente di esprimere un giudizio binario su ciascun film | 2 |
| RF03 | Il sistema permette all’utente di ricercare tramite keyword un film sul titolo del film indipendentemente dalla piattaforma di streaming | 3 |
| RF04 | Il sistema deve permettere all’utente di aggiungere un film alla watchlist | 4 |
| RF05 | Il sistema deve permettere all’utente di aggiungere un film alla lista watchLater | 5 |
| RF06 | Il sistema deve permettere all’utente di visualizzare la watchLater | 6 |
| RF07 | Il sistema deve permettere all’utente di visualizzare la watchLater | 7 |

## Requisiti sui dati

|  |  |  |
| --- | --- | --- |
| ID | Requisito | Origine (n. frase dei requisiti revisionati) |
| RD01 | Un film è caratterizzato da un titolo, da una descrizione,da una durata, da un anno di rilascio,da un rating, da un voto imdb e da una serie di piattaforme su cui è disponibile(anche nessuna) | 9 |

## Modellazione dei casi d’uso

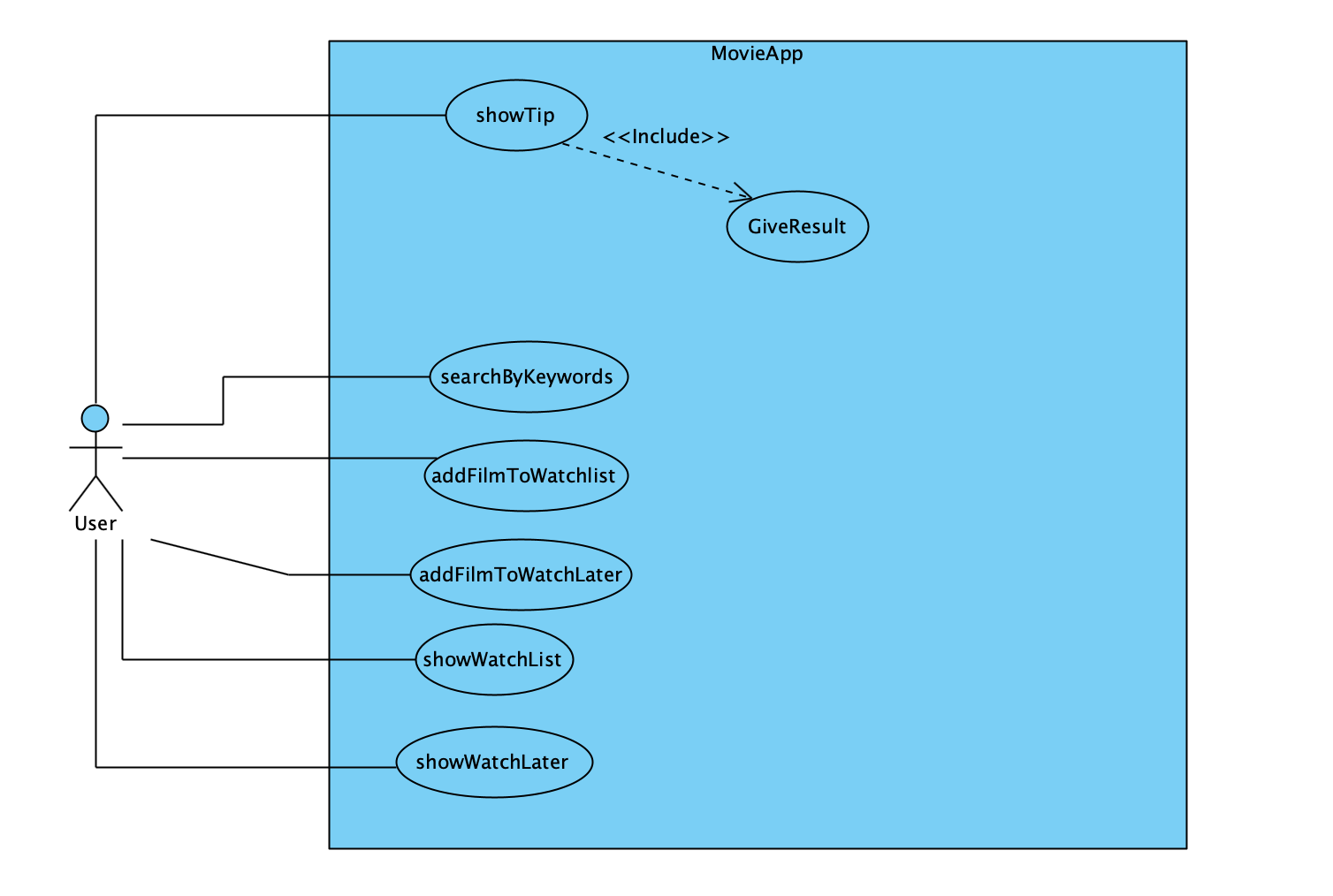
## Attori e casi d’uso

***Attori Primari:***

* Utente

|  |  |  |  |
| --- | --- | --- | --- |
| Caso d’uso | Attori Primari | Attori Secondari | Incl. / Ext. |
| showTip | Utente | … | GiveResult |
| searchByKeywords | Utente | … | … |
| addFilmToWatchlist | Utente | … | … |
| addFilmToWatchLater | Utente | … | … |
| showWatchList | Utente | … | … |
| showWatchLater | Utente | … | … |

## Diagramma dei casi d’uso



## Scenari

|  |  |
| --- | --- |
| Caso d’uso: | showTip |
| Attore primario | Utente |
| Attore secondario | --- |
| Descrizione | L’utente indica che vuole ricevere un tip, il sistema consiglia un film |
| Pre-Condizioni |  |
| Sequenza di eventi  principale | 1. L’utente comunica che vuole ricevere un consiglio  2. il sistema visualizza un film come consiglio |
| Post-Condizioni | L’utente visualizza il consiglio |
| Casi d’uso correlati | *giveResult* |
| Sequenza di eventi  alternativi |  |

|  |  |
| --- | --- |
| Caso d’uso: | giveResult |
| Attore primario | Utente |
| Attore secondario | --- |
| Descrizione | L’utente espirme il suo giudizio binario su un film |
| Pre-Condizioni | L’utente ha visualizzato un film tramite showTip |
| Sequenza di eventi  principale | 1. l’utente esprime un giudizio binario sul film visualizzato |
| Post-Condizioni | Il giudizio binario viene memorizzato |
| Casi d’uso correlati |  |
| Sequenza di eventi |  |

|  |  |
| --- | --- |
| Caso d’uso: | searchByKeywords |
| Attore primario | Utente |
| Attore secondario | --- |
| Descrizione | Il sistema permette all’utente di ricercare tramite i film tramite keyword |
| Pre-Condizioni |  |
| Sequenza di eventi  principale |  |
| Post-Condizioni |  |
| Casi d’uso correlati |  |
| Sequenza di eventi  alternativi |  |

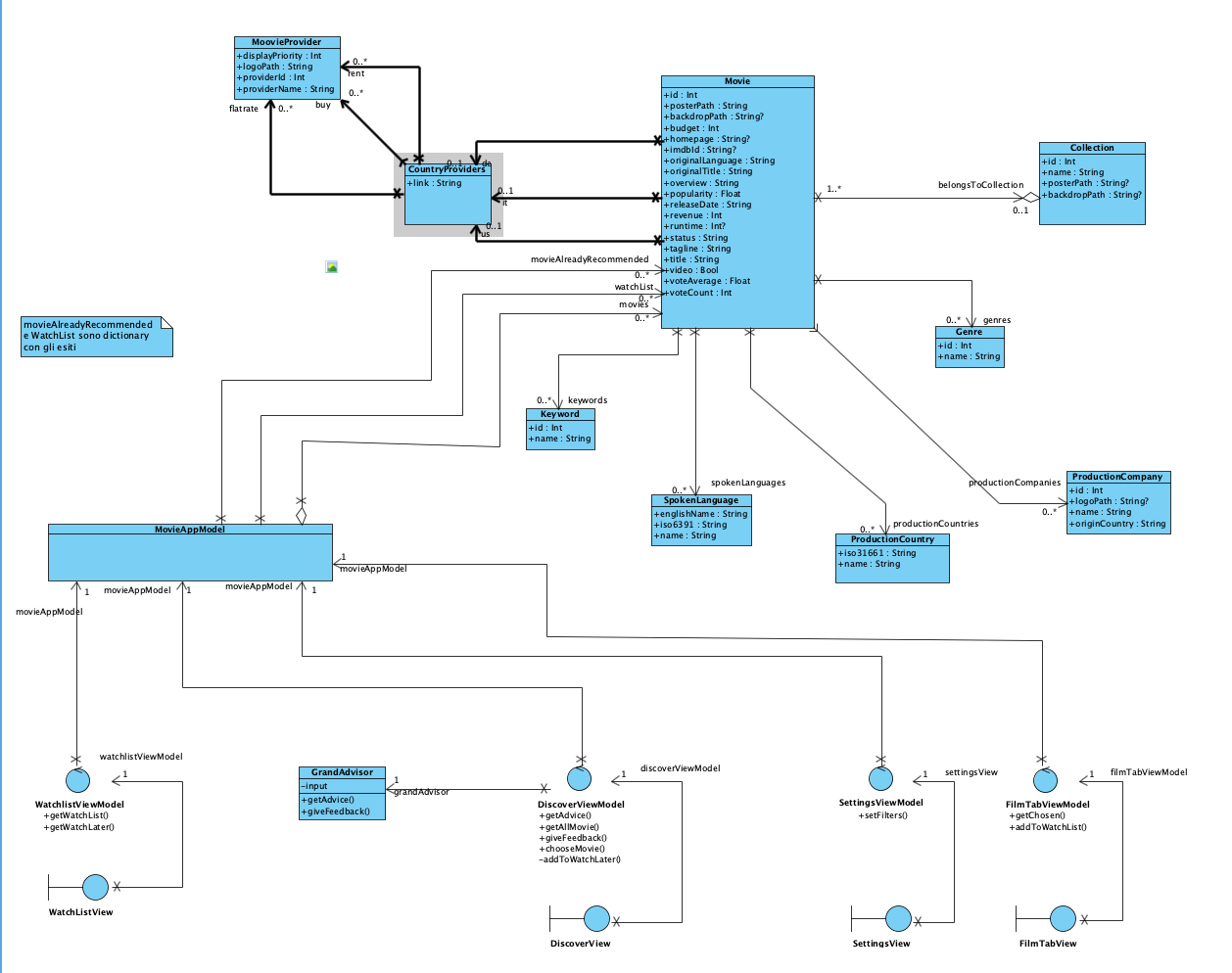
|  |  |
| --- | --- |
| Caso d’uso: | addFilmToWatchlist |
| Attore primario | Utente |
| Attore secondario | --- |
| Descrizione | L’utente aggiunge un film alla watchlist |
| Pre-Condizioni |  |
| Sequenza di eventi  principale | 1. L’utente comunica al sistema di voler aggiungere un film alla watchlist 2. Il sistema memorizza il film nella watchlist |
| Post-Condizioni | Il sistema memorizza la watchlist |
| Casi d’uso correlati |  |
| Sequenza di eventi  alternativi |  |

|  |  |
| --- | --- |
| Caso d’uso: | addFilmToWatchlater |
| Attore primario | Utente |
| Attore secondario |  |
| Descrizione | L’utente aggiunge un film alla watchlater |
| Pre-Condizioni | Il rider è registrato nel sistema  Il cliente ha completato l’ordinazione |
| Sequenza di eventi  principale | 1. L’utente comunica al sistema di voler aggiungere un film alla watchlater 2. Il sistema memorizza il film nella watchlater |
| Post-Condizioni |  |
| Casi d’uso correlati |  |
| Sequenza di eventi  alternativi |  |

|  |  |
| --- | --- |
| Caso d’uso: | showWatchList |
| Attore primario | Utente |
| Attore secondario |  |
| Descrizione | L’utente comunica di voler visualizzare la watchlist |
| Pre-Condizioni |  |
| Sequenza di eventi  principale | 1. L’utente comunica di voler visualizzare la watchlist 2. Il sistema mostra la watchlist |
| Post-Condizioni | Il sistema memorizza i dati dell’avvenuta consegna |
| Casi d’uso correlati |  |
| Sequenza di eventi  alternativi |  |

|  |  |
| --- | --- |
| Caso d’uso: | showWatchLater |
| Attore primario | Utente |
| Attore secondario |  |
| Descrizione | L’utente comunica di voler visualizzare la watchlater |
| Pre-Condizioni |  |
| Sequenza di eventi  principale | 1. L’utente comunica di voler visualizzare la watchlater 2. Il sistema mostra la watchlater |
| Post-Condizioni |  |
| Casi d’uso correlati |  |
| Sequenza di eventi  alternativi |  |

## Diagramma delle classi



## Diagrammi di sequenza

Riportare i diagrammi di sequenza di analisi per i casi d’uso sviluppati fino alla codifica in Java.

## Verifica della completezza dei requisiti

Si suggerisce allo studente di verificare che tutti i requisiti informali siano rappresentati nel modello UML e/o negli scenari. A tale scopo, lo studente può elencare i requisiti (funzionali, requisiti sui dati, altri requisiti) riportando per ciascun requisito gli elementi dei diagrammi UML con i quali è rappresentato nel modello di analisi. Dopo l’elencazione, lo studente verifichi che tutti i requisiti siano stati modellati nei diagrammi UML.

# Piano di test funzionale

Progettare i casi di test funzionale con la tecnica del *Category Partition Testing*. Descrivere il procedimento di calcolo.

**PIANO DI TEST UTILIZZANDO IL METODO DEL *CATEGORY-PARTITION TESTING* PER LA FUNZIONALITÀ “*.......*”.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Categoria...** | **Categoria...** | **Categoria...** | **Categoria...** | **Categoria...** |
| * ... * ... * ... | * ... * ... * ... | * ... * ... * ... | * ... * ... * ... | * ... * ... * ... |

**TEST SUITE**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Descrizione** | **Classi di equivalenza coperte** | **Pre-condizioni** | **Input** | **Output Attesi** | **Post-condizioni Attese** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

# Progettazione

## Diagramma delle classi

Riportare il diagramma delle classi di progettazione. Reificare eventuali classi associative del diagramma delle classi di analisi. Specificare argomenti e tipo di ritorno delle operazioni (per quelle più significative, coinvolte nei casi d’uso sviluppati fino alla implementazione). Includere classi del dominio della soluzione, come strutture dati e classi DAO. Raggruppare le classi in package.

## Diagrammi di sequenza

Riportare i diagrammi di sequenza di progetto per il/i casi d’uso sviluppati fino alla codifica in Java.

# Implementazione

Non includere il codice sorgente, ma descrivere l’implementazione in Java, descrivendo gli artefatti di codifica:

* Elencare:
  + package, classi, tipi di eccezione definiti
* Elencare gli artefatti necessari per l’installazione ed esecuzione del programma, senza ovviamente l’ambiente di sviluppo come Eclipse (DB h2, eventuali librerie e versioni di Java che l’utilizzatore deve avere installati, file .class, .jar, …)
* Produrre un eventuale diagramma di deployment
* Eventualmente inserire la documentazione del codice prodotta con Javadoc (relativamente alle funzionalità implementate)
* Riportare il numero di LOC e di LLOC scritte in Java
* Confrontare con la stima dei costi effettuata e commentare eventuali scostamenti

# Testing

## Test strutturale

## Complessità ciclomatica

Costruire il Control Flow Graph per uno o due dei metodi delle classi implementate (si scelgano metodi non proprio banali), e:

- si mostri il calcolo del numero ciclomatico;

- si indichino i percorsi linearmente indipendenti;

Prima o a fianco del CFG riportare il codice Java del metodo.

Es.:

public static boolean CalcolaStatistica(String stringaData, int CAP, boolean scelta) {

if(scelta)

stringaData = "01/"+stringaData;

LocalDate data = null;

DateTimeFormatter dateTimeFormatter = DateTimeFormatter.ofPattern("dd/MM/yyyy");

try {

if(!(MonitoraggioAmbiente.getInstance().VerificaCAP(CAP))) { //Verifica CAP

System.err.println("CAP non valido");

return false;

}

}

catch(IOException e) {

System.err.println("Errore di I/O durante il controllo del CAP" + e.getMessage());

return false;

}

try {

data = LocalDate.parse(stringaData, dateTimeFormatter); //Dalla sequenza data.parse() != null

if(data==null) { //Verifica DATA

System.err.println("Data non valida");

return false;

}

}catch(DateTimeParseException e) {

System.err.println("Data non valida");

return false;

}

java.util.Map<String, Float> Dati = null;

try {

if(scelta) {

Dati = MonitoraggioAmbiente.getInstance().CalcolaStatisticaMese(CAP, data);

System.out.println("Statistica del mese inserito:\n");

}

else {

Dati = MonitoraggioAmbiente.getInstance().CalcolaStatisticaGiorno(CAP, data);

System.out.println("Statistica del giorno inserito:\n");

}

for(String key:Dati.keySet()) {

System.out.println(key +":"+(key.length()<14?"\t\t":"\t") + Dati.get(key)); //Output

}

} catch (StatisticaException e) {

System.err.println(e.getMessage()); return false;

} catch (MisuraException e) {

System.err.println(e.getMessage()); return false;

}

return true;

}

}

Control Flow Graph



NUMERO CICLOMATICO:

numero di regioni chiuse del grafo = 6

numero di nodi predicati (0,2,4,6,10) +1 = 6

# archi – # nodi + 2 = (15 - 11) + 2 = 6

CAMMINI:

1. 0-1-2-3
2. 0-1-2-4-5
3. 0-1-2-4-6-7-9
4. 0-1-2-4-6-8-9
5. 0-1-2-4-6-7-9-10-9
6. 0-2-4-6-7-9

## Test funzionale

Descrivere i risultati dell’esecuzione dei test funzionali precedentemente pianificati adoperando lo schema di tabella seguente.

Descrivere le eventuali azioni di *debugging* a seguito di casi di test con esito *FAIL*.

Commentare se gli eventuali difetti rilevati dal test funzionale potevano essere rilevati anche da un test strutturale.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Descrizione** | **Classi di equivalenza coperte** | **Pre-condizioni** | **Input** | **Output Attesi** | **Post-condizioni Attese** | **Output Ottenuti** | **Post-condizioni Ottenute** | **Esito**  **(*FAIL*, *PASS*)** |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |