**Project Specification Document**

<<Company Name>>

<< Team Member >>

<< Team Member >>

<< Team Member >>

<< Team Member >>

<< Team Member >>

Table of Contents

Section Page

[Table of Contents ii](#_Toc372915361)

[1. Project Vision and Objectives 1](#_Toc372915362)

[1.1 Project Scope and Vision 1](#_Toc372915363)

[1.2 Project Goals and Objectives 1](#_Toc372915364)

[2. Project Planning 2](#_Toc372915365)

[2.1 Project Lifecycle 2](#_Toc372915366)

[2.2 Project Setup 2](#_Toc372915367)

[2.3 Stakeholders 2](#_Toc372915368)

[2.4 Project Resources 3](#_Toc372915369)

[2.5 Assumptions 3](#_Toc372915370)

[2.6 Estimates 3](#_Toc372915371)

[3. Project Tracking 5](#_Toc372915372)

[3.1 Tracking 5](#_Toc372915373)

[3.2 Communication Plan 5](#_Toc372915374)

[Regularly Scheduled Meetings 5](#_Toc372915375)

[Information To Be Shared Within Our Group 5](#_Toc372915376)

[Information To Be Provided To Other Groups 6](#_Toc372915377)

[Information Needed From Other Groups 6](#_Toc372915378)

[3.3 Deliverables 6](#_Toc372915379)

[3.4 Project Metrics 6](#_Toc372915380)

[4. Requirements 8](#_Toc372915381)

[4.1 Overall Description 8](#_Toc372915382)

[4.2 Users and Roles 8](#_Toc372915383)

[4.3 Use Case Diagrams 8](#_Toc372915384)

[4.4 High-Level Requirements 8](#_Toc372915385)

[4.5 Low-Level Requirements 9](#_Toc372915386)

[4.6 Constraints and Limitations 10](#_Toc372915387)

[5. Design 11](#_Toc372915388)

[5.1 Introduction 11](#_Toc372915389)

[5.2 Scope 11](#_Toc372915390)

[5.3 High-Level Component Design 11](#_Toc372915391)

[5.4 Activity Diagrams 11](#_Toc372915392)

[5.5 Class Diagram 11](#_Toc372915393)

[5.6 Sequence Diagram 12](#_Toc372915394)

[5.6 Data Architecture 12](#_Toc372915395)

[5.7 Alternative Designs and Design Rationale 12](#_Toc372915396)

[5.8 Low-Level Component Design 12](#_Toc372915397)

[Class 1 12](#_Toc372915398)

[Description 12](#_Toc372915399)

[Variables 12](#_Toc372915400)

[Methods 13](#_Toc372915401)

[Class 2 13](#_Toc372915402)

[Description 13](#_Toc372915403)

[Variables 13](#_Toc372915404)

[Methods 13](#_Toc372915405)

[6. User Interface 15](#_Toc372915406)

[6.1 UI Description 15](#_Toc372915407)

[6.2 UI Mockup 15](#_Toc372915408)

[7. Test Plan 16](#_Toc372915409)

[7.1 Test Plan Description 16](#_Toc372915410)

[7.2 Testing Tools 16](#_Toc372915411)

[7.3 Test Data 16](#_Toc372915412)

[7.4 Test Types and Frequency 16](#_Toc372915413)

[7.5 Test Coverage 16](#_Toc372915414)

[7.6 Test Location 17](#_Toc372915415)

[8. Project Closure 18](#_Toc372915416)

[8.1 Goals / Vision 18](#_Toc372915417)

[8.2 Delivered Solution 18](#_Toc372915418)

[8.3 Remaining Work 18](#_Toc372915419)

[9. Deliverables 19](#_Toc372915420)

[9.1 Study Results 19](#_Toc372915421)

[9.2 Requirements and Design Documents 19](#_Toc372915422)

[9.3 Code 19](#_Toc372915423)

[9.4 Tests and Test Results 19](#_Toc372915424)

[9.5 Build Process Documents 19](#_Toc372915425)

[9.6 Install Process Documents 19](#_Toc372915426)

[9.7 Administrator’s and/or User’s Manual 19](#_Toc372915427)

[9.8 Postmortem Document 19](#_Toc372915428)

[9.9 Final Report 19](#_Toc372915429)

[10. Definitions and Acronyms 20](#_Toc372915430)

1. Project Vision and Objectives

1.1 Project Scope and Vision

<< Provide a concise statement on the purpose of the project; the problem or opportunity addressed. The explanation should include what you intend to do and by when. Vision – how will the customer’s world improve as a result of this project? When appropriate, tie this into what is currently being considered or has just been completed at the company. >>

1.2 Project Goals and Objectives

<< State, in quantifiable terms, if possible, the goals and objectives of the project. Goals may be related to product, process, quality, or teamwork. >>

|  |  |
| --- | --- |
| **#** | **Goal or Objective** |
| 1 | Make the system extensible – future updates like xxx can be done easily |
| 2 | Make the system easy to support – provide good documentation, configuration/build files, administrator’s manual |
| 3 | Make the system very easy to use – users would agree that minimal to no training is needed |
| 4 | Build a prototype that demonstrates the user interface by xx/xx/xx - in order to get early feedback from the customer/users |
| 5 | Have fun working on the project |

2. Project Planning

2.1 Project Lifecycle

<< Describe the lifecycle of the project. You can choose from an existing lifecycle definition or create your own. >>

<< **Example:** The team will use a waterfall approach. Our team will not begin a new phase of development (i.e. design, coding, testing, etc.) until the previous phase has been completed. A working prototype will not be created, but any part of the project that has been completed by midterm will be demonstrated at the review. >>

<< **Example:** The team will use a spiral approach. Our team will make an initial pass through the requirements and design phases so that an early prototype can be created by midterm. The prototype will serve as a baseline for the final release and any releases that occur between the initial prototype and the final release. >>

<< **Example:** The team will use an agile approach. Our team will gather requirements and create a high level development plan at the onset of the project and then implement the gathered requirements over three iterations. The team will follow a SCRUM-like approach with an emphasis on frequent meetings and collaboration. >>

2.2 Project Setup

<< Define some of the basic project decisions that will be used on this project. >>

|  |  |
| --- | --- |
| **#** | **Decision Description** |
| 1 | Windows 7, C#, OpenSphere, will be used on the project. |
| 2 | Standards that must be followed include the default Capstone coding standard. |
| 3 | Special access privileges needed, nondisclosure forms, must be taken care of. |
| 4 | A virtual server image will be set up at NDSU that matches the customer environment (image provided by customer) |
|  |  |

2.3 Stakeholders

<< Identify all stakeholders for this project (groups or individuals that are affected by or are in some way accountable for the outcome of the project – business managers, end users, developers, testers, support people, instructors, etc.) >>

|  |  |
| --- | --- |
| **Stakeholder** | **Role** |
| Person A | Sponsor |
| Person B | Mentor |
| Person C | Instructor |
| Person D | Team member |
| Person E | Team member |
| Person F | Team member |

2.4 Project Resources

<< Identify the anticipated resources required for this project. This can include staff members who will work on the project, equipment needed for the project, special software that will need to be acquired, or any other resource necessary for the project. >>

|  |  |
| --- | --- |
| **Resource** | **Resource Description** |
| Database Server | A database server provided by the sponsoring company. |
| Capstone Team | Our team of students who will be the primary developers of the project. |
| Jim Somebody | The mentor who will be able to provide us with technical assistance. |
| Mac Workstation | An OS X workstation with X Code for developing the OS X version of the software. |
| Android Phone | An Android phone to be used as test hardware for the mobile version of the software. |

2.5 Assumptions

**<<**State any assumptions upon which the project is based. Assumptions may be related to staffing, resources, tools, and schedules/deadlines. >>

|  |  |
| --- | --- |
| **#** | **Assumption** |
| A1 | The sample data will be available by March 15th. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

2.6 Estimates

<< Provide estimates for the size of the project. This includes estimates about the total hours to be spent on the project, the amount of code written, the amount of supporting or additional documentation written, and any other estimates which might help the project team to consider the amount of effort that will go into the product.

These are high-level estimates that are to be made during the initial planning phase of the project and should not be modified thereafter. After the project is completed, add in the actual values (which should be available in the Microsoft Project document) for all metrics. It is not important that the original estimates closely match the actual values, but should be useful in informing future estimates. >>

|  |  |  |
| --- | --- | --- |
| **Metric** | **Estimated Value** | **Actual Value** |
| Hours | 250 hours |  |
| Lines of Code | 1500 lines |  |
| Manual / Help Document Pages | 10 pages |  |
| Unit Tests | 100 tests |  |
|  |  |  |

3. Project Tracking

3.1 Tracking

<< Provide information about how the project will be tracked and where information will be kept. This should include information such as what type of source control is being used and how it can be accessed, any bug-tracking system that will be used for the project and where it can be accessed, what type of regressing testing suite will be used and where it can be accessed, and any similar information that provides details on the project’s status, etc. >>

|  |  |  |
| --- | --- | --- |
| **Information** | **Description** | **Link** |
| Code Storage | Project code will be stored in a git-hub account. | [Link](http://www.yourlinkhere.com/) |
| Bug Tracking | Bug tracking will be done with Trac. | [Link](http://www.yourlinkhere.com/) |
| Project Schedule | The project schedule will be stored in the git repository. | [Link](http://www.yourlinkhere.com/) |
| Continuous Integration | Continuous integration will be done with Jenkins. | [Link](http://www.yourlinkhere.com/) |
| Regression Testing | Regression testing will use JUnit unit tests and Jenkins. | [Link](http://www.yourlinkhere.com/) |
|  |  |  |

3.2 Communication Plan

<< Identify all communications you will provide to other groups and all communications you need to receive from other groups. Share this information with affected groups. Verify that all stakeholders are included. >>

Regularly Scheduled Meetings

|  |  |  |
| --- | --- | --- |
| Meeting Type | Frequency/Schedule | Who Attends |
| Conference Call | Bi-weekly | Project team and mentor |
| Team Meeting | Weekly | Project team |
| Short Meeting | Weekly in class | Project team |
| Sprint Planning Meeting | Start of each sprint | Project team and mentor |
| Sprint Retrospective Meeting | End of each sprint | Project team |
| Sprint Review Meeting | End of each sprint | Project team, mentor, and sponsor |

Information To Be Shared Within Our Group

|  |  |  |  |
| --- | --- | --- | --- |
| Who? | What Information? | When? | How? |
| Project team | Task assignments | Weekly | Team meetings, listing in MS Project file. |
|  |  |  |  |

Information To Be Provided To Other Groups

|  |  |  |  |
| --- | --- | --- | --- |
| Who? | What Information? | When? | How? |
| Sponsor and mentor | Final report | At completion of project | Req./Design docs., code, Power Point presentation |
| Sponsor and mentor | Project baselines | At the end of each sprint | Onsite customer demo, access to repository |
| Sponsor and mentor | Weekly report | Weekly | Email and Trac site access |

Information Needed From Other Groups

|  |  |  |  |
| --- | --- | --- | --- |
| Who? | What Information? | When? | How? |
| Sponsor and mentor | Requirement changes | Start of each sprint | Conference call or meeting with sponsor and mentor. |
| Adam Helsene | Availability of test server | Start of second sprint | Email |

3.3 Deliverables

<< Identify the major deliverables that this project is expected to produce. Assume the deliverables apply to all features or stories listed above unless indicated otherwise. Deliverables may include prototypes, requirements, designs, documentation, software, etc. Identify whether the item is an “**interim**” deliverable (a deliverable that is required to complete this project, but is not part of the final, released product) or a “**final**” deliverable (a deliverable that is part of the final product). >>

|  |  |  |  |
| --- | --- | --- | --- |
| # | Deliverable | Delivery Time | |
|  |  | Interim | Final |
| 1 | Prototype Code |  |  |
| 2 | Build/Configuration Files |  |  |
| 3 | Regression Tests |  |  |
| 4 | Project Documentation |  |  |
| 5 |  |  |  |
| 6 |  |  |  |

3.4 Project Metrics

<< This section identifies the standard metrics to be collected in your project. Information can be collected in a spreadsheet or using other means. By default, most will be recorded in the MS Project Plan.

List all metrics that will be collected, how often they will be collected, and where the metrics will be recorded. >>

|  |  |  |
| --- | --- | --- |
| Metric | Frequency | Location |
| Estimated Effort (in hours) | Per task | MS Project Plan |
| Actual Effort (in hours) | At task completion | MS Project Plan |
| Maintenance Effort (in hours) | As occurring | MS Project Plan |
| Estimated Size (in LoC) | Per task requiring code | MS Project Plan |
| Actual Size (in LoC) | At completion of task requiring code | MS Project Plan |
|  |  |  |

4. Requirements

4.1 Overall Description

<< Provide a more detailed, two to three paragraph description of the project. This description may include more technical details to describe the purpose of the project. >>

4.2 Users and Roles

<< Provide a list and description of the different types of users or roles within the system. This may include different classes of users, such as administrator, instructor, student, etc. This list may also include autonomous agents that interact with the system as well. These may include users (or personas) that are used as part of any user stories produced for the project. >>

|  |  |
| --- | --- |
| **User** | **Description** |
| User | A regular user who is the primary operator of the system. |
| Admin | A user who also has the ability to assign user privileges to other users. |
|  |  |
|  |  |
|  |  |

4.3 Use Case Diagrams

<< Provide any use-case diagrams that are being used as part of the project. Uniquely label each use case so that if necessary it is easy to reference from other parts of the document. >>

4.4 High-Level Requirements

<< This section lists the high-level requirements for the project and information about which of these requirements the team has committed to completed (as opposed to those which the team will not commit to, or is targeting for completion, but cannot guarantee) and when those requirements were added to the project.

The **ID** column provides a high-level ID for each requirement. This is useful for generating low-level requirement IDs which can be easily associated with a particular high-level requirement. The **Added** column lists when the requirement was added to the project as it is possible that not all requirements will be elicited or available at the project’s onset. The **Description** column gives a description of the high-level requirement. The **Status** column indicates whether these high-level requirements are committed (i.e. will be completed by the team), targeted (i.e. will be completed if the team has time after all committed requirements are completed), or not committed (i.e. will not be completed by the team).

High level requirements may also be user stories, from which multiple low-level requirements may be derived. If user stories are being used >>

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Added** | **Description** | **Status** |
| 100 | Onset | Users may view 3D representations of nodes on a globe | **C** |
| 200 | Onset | Users may view the movement of nodes over time | **C** |
| 300 | Onset | Users may view available data about a node | **C** |
| 400 | Onset | Lines of communication display relationships between nodes | **C** |
| 500 | Onset | On-click zoom functionality available to users | **C** |
| 600 | Onset | Rewind/Fast Forward/Pause/Play allows movement of nodes to be easily visualized over time | **C** |
| 700 | Onset | User interface is elegant and easy to use | **C** |
| 800 | Onset | Add halo effect around nodes | **T** |
| 900 | Onset | User may filter out different types of nodes | **C** |
| 1000 | Onset | Line segment denoting previous locations of nodes | **C** |
| 1100 | Sprint 1 | Design 3D models | **T** |
| 1200 | Sprint 2 | 3D models face the correct direction when moving | **T** |
| 1300 | Sprint 3 | 3D models are painted | **NC** |

4.5 Low-Level Requirements

<< This section lists low-level requirements for each of the high-level requirements. Requirements in this section should use the formal descriptive language for requirement specification (i.e. must/shall language). Additionally, some thought should be given to how these requirements will be verified and a brief description of how the requirement will be tested should be provided.

The **ID** column specifies the ID for the low-level requirement. This can be used to refer to the requirement in other documentation (e.g., the MS Project Plan) and to provide traceability. The **Description** column specifies the requirement formally. The **Verification** column provides information about how the requirement will be tested.

Remember that not all requirements are necessarily functional. Requirements can also specify requirements for security (e.g. who can and cannot use a system), globalization (e.g. descriptions and other strings will not be hard coded so that the software can be translated into other languages more easily), portability (e.g. developing an iPhone to be compatible with other smart phone platforms like Android or Windows Phone) accessibility (e.g. the program complying with ADA guidelines), availability (e.g. the system will be able to reboot in under 2 minutes in the event of failure.), etc. >>

|  |  |  |
| --- | --- | --- |
| **ID** | **Description** | **Verification** |
| 110 | Nodes shall be represented in three-dimensional space. |  |
| 120 | Users shall be able to manipulate the map to view nodes from different perspectives. | Usability test to determine if map can be manipulated in all three dimensions. |
| 130 | Node positional information shall conform to guidelines established in IEEE 702.34 guidelines. | Create test cases to verify that node positional data is stored in accordance with given standards. |
| 210 | Node information shall be updated at an interval of at least 10 minutes. |  |
| 220 | Users shall be able to determine if new node information has not been updated. | Provide indication that node information is fresh for last ten minutes. Alert user when node information is not fresh. |
| . |  |  |
| . |  |  |
| . |  |  |
| 710 | Strings for field descriptions in the user interface shall not be hard coded. |  |
| 720 | Alternative language packs shall be selectable by the user. | Create stub test data to simulate alternative languages. |

4.6 Constraints and Limitations

<< This section provides a list of constraints and limitations for the project. This provides additional information about any limitations that may exist in the project (e.g. will not work in versions of Internet Explorer prior to 8.0) that are not covered by requirements, but provide important information related to the project. This may include constraints related to security, performance, and other aspects of the system.

The **Constraint** field lists the constraint or limitation for the project. The **ID** field lists the related requirement ID (if any) that provides additional context for the constraint. >>

|  |  |
| --- | --- |
| **Constraint** | **ID** |
| Team will not provide alternative language pack files. | 720 |
| Provided 3D models will not be of professional-grade quality. | 1100 |
| Team will use Visual Studio 2007 for development as 2010 is not available. |  |
|  |  |
|  |  |

5. Design

5.1 Introduction

<< Briefly describe the major aspects of the design and, if applicable, how a developer will use it. For example: “Create and post a General Ledger transaction using the glTrx routines. Perform account inquiries with gjJournal routines.”

5.2 Scope

<< Provide a brief overview of the scope of this design. Also touch on anything that will not be included in this document. >>

5.3 High-Level Component Design

<< Create a diagram of the high-level components or modules in the program, linking them with arrows to show any dependencies. Also complete the tables to provide a description of each module as well as the table which traces components to their related requirements. >>

|  |  |
| --- | --- |
| **Component** | **Description** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Component** | **Related Requirements** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

5.4 Activity Diagrams

<< Include activity diagrams for important workflows in the program. At least one diagram should be included for the main workflow in the program. Optionally include labels that indicate which component is responsible for that part of the activity. Activity diagrams for components which perform complex tasks should also be included. >>

5.5 Class Diagram

<< Include a class diagram for all classes to be designed. Optionally include major data elements of those classes and important methods and functions that will be used by other classes. >>

5.6 Sequence Diagram

<< Include sequence diagrams for important functionality of the program to indicate control flow. These diagrams should include classes found in the class diagram and use the methods for those classes to show the interaction between them. >>

5.6 Data Architecture

<< Include any information or diagrams that provide details about databases, xml configuration files, or other data structures that are a part of the system. If a very specific format is required, it may be worthwhile to provide a more robust description or a detailed design such as a database schema. >>

5.7 Alternative Designs and Design Rationale

<< Provide information regarding other designs which were considered, but not chosen. Provide rationale for why the chosen design was selected over the alternative candidate designs. >>

5.8 Low-Level Component Design

<< Optionally provide a detailed design for each class or module in the program. Include information about the variables, methods, and other important aspects of the class such as exception handling.

Note that the templates provided here are designed for Java and that some languages will not contain >>

Class 1

Description

|  |  |  |
| --- | --- | --- |
|  | Access | Description |
|  |  |  |
| **Extends** |  | |
| **Implements** |  | |
| **Throws** |  | |

Variables

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Access | Description |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Methods

|  |  |  |  |
| --- | --- | --- | --- |
| Method Name | Return Type | Access | Description |
|  |  |  |  |
| **Parameter** | **Type** | **Description** | |
|  |  |  | |
|  |  |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| Method Name | Return Type | Access | Description |
|  |  |  |  |
| **Parameter** | **Type** | **Description** | |
|  |  |  | |
|  |  |  | |

Class 2

Description

|  |  |  |
| --- | --- | --- |
|  | Access | Description |
|  |  |  |
| **Extends** |  | |
| **Implements** |  | |
| **Throws** |  | |

Variables

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Access | Description |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Methods

|  |  |  |  |
| --- | --- | --- | --- |
| Method Name | Return Type | Access | Description |
|  |  |  |  |
| **Parameter** | **Type** | **Description** | |
|  |  |  | |
|  |  |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| Method Name | Return Type | Access | Description |
|  |  |  |  |
| **Parameter** | **Type** | **Description** | |
|  |  |  | |
|  |  |  | |

6. User Interface

6.1 UI Description

<< Provide a brief description of the UI that will be used in this program and how users will interact with the program. >>

6.2 UI Mockup

<< Create a mockup of the user interface. This can be a simple drawing that demonstrates key parts of the user interface or a screenshot of a prototype created within an IDE. >>

7. Test Plan

7.1 Test Plan Description

<< Provide a brief description of how testing will be conducted for this project. >>

7.2 Testing Tools

<< Provide a brief description of any testing tools, suites, etc. that will be used to assist with testing the project’s code. >>

|  |  |
| --- | --- |
| **Tool** | **Description / Function** |
| Jenkins | Continuous integration and regression testing. |
|  |  |
|  |  |

7.3 Test Data

<< Identify any data that will be necessary as part of the test process. >>

|  |  |  |
| --- | --- | --- |
| **Data** | **Description** | **Link** |
| Sample Data | Sample data of known good and bad examples construct unit tests. | [Link](http://examplelink.com) |
| User Database | Access to company’s Active Directory to test authentication for users. | [Link](http://examplelink.com) |
| Order Database | Stub data for unit tests. | [Link](http://examplelink.com) |

7.4 Test Types and Frequency

<< Provide a list of all the different types of tests that will be performed as part of this project, a brief description of each, and the frequency with which each test will be conducted. >>

|  |  |  |
| --- | --- | --- |
| **Type** | **Description** | **Frequency** |
| Integration Test | Integration test of existing code | Weekly starting with Sprint 3 |
| Regression Test | Run unit tests against all code | Daily (2:00 AM) |
| Usability Test | Perform manual usability test of UI. | Every Sprint |

7.5 Test Coverage

<< Identify components or other aspects of the project that will not be tested and provide rationale for that reasoning. >>

|  |  |
| --- | --- |
| **Component** | **Rationale** |
| Req. 720 | Do not have sufficient time to generate stub data. |
|  |  |
|  |  |

7.6 Test Location

<< Indicate where unit tests, test cases, etc. are being stored >>

8. Project Closure

8.1 Goals / Vision

**<<** Provide an update to the vision statement that was originally stated in the Project Initiation document. >>

8.2 Delivered Solution

<< Provide a high-level description of what was planned and what is being delivered. >>

8.3 Remaining Work

<< Provide a short summary of what should be done next, ways of further improving the project, or any additional recommendations. >>

9. Deliverables

<< This section contains information on the location of any deliverables for the project. If there are none for a particular category, indicate that it is not applicable. Otherwise provide the name of any files, etc. and where they can be found. >>

9.1 Study Results

<< The location of the results of any studies performed as part of this project. >>

9.2 Requirements and Design Documents

<< The location of any requirements and design documents. >>

9.3 Code

<< The location of any code written for this project. >>

9.4 Tests and Test Results

<< The location of any tests (unit, regression, etc.), test results, or other testing documents. >>

9.5 Build Process Documents

<< The location of any documents detailing build processes. >>

9.6 Install Process Documents

<< The location of any documents describing installation processes. >>

9.7 Administrator’s and/or User’s Manual

<< The location of any manuals, or help documentation. >>

9.8 Postmortem Document

<< The location of the Postmortem document. >>

9.9 Final Report

<< The location of the Final Report document. >>

10. Definitions and Acronyms

<< This section provides a definition for terms or acronyms used in this document which may not be familiar for all users. >>

|  |  |
| --- | --- |
| **Term** | **Definition** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |