**Detailed Design Document for “Think Tank”**

Team Members:

David Gershuni, Shiyao Wu, Kaiwen Shi, Woye Lin, Jon Koehmstedt, Emily Pham

We will implement “Think Tank” using Java and the *libgdx* Java game development library. Our target hardware platforms are desktop PCs running Linux, Mac, or Windows. We chose libgdx because it is a powerful library for developing high performance games in Java that reduces the need for boilerplate code and provides useful high-level constructs such as actors, cameras, renderers, sprites, and more.

ThinkTankClient: Class extends gdx.Game

(Top level class on client side that maintains the connection to server and the stage of the game)

- server: ServerSocket (used by client to send messages to server)

- client: Socket (used by client to receive messages from server)

- stage: Stage

DesktopLauncher: Class (launches the client application)

* main (String args[])

Team: Class

- players: ArrayList<Player>

-  brain: Brain

- teamID: int

Player: Class

- tank: Tank

- name: String

           - password: String

Brain: Class extends Actor

- health: int (health of base from 0-1000)

- grayMatter: int

GameMap: Class

- camera: Orthographic Camera

- map: TiledMap

- mapRenderer: OrthogonalTiledMapRenderer

BattleMap: Class extends GameMap

- camera: gdx.OrthographicCamera

- map: gdx.TiledMap

- mapRenderer: gdx.OrthogonalTiledMapRenderer

MiniMap: Class extends GameMap

- camera: gdx.OrthographicCamera

- map: gdx.TiledMap

- mapRenderer: gdx.OrthogonalTiledMapRenderer

UtilityBar: Class

(Contains Upgrade Buttons and Gray Matter indicator)

- weaponsUpgradeButton: gdx.ImageButton

- shieldUpgradeButton: gdx.ImageButton

- turretButton: gdx.ImageButton

- grayMatter: gdx.Label

- healthBar: gdx.ProgressBar

- bulletType: gdx.Image

GameStateBar: Class

- team1Health: gdx.ProgressBar

- team2Health: gdx.ProgressBar

- gameTime: gdx.Label

ChatWindow: Class

- chatLog: gdx.TextArea (Scrollable)

- chatTF: gdx.TextField

- sendButton: gdx.Button

Tank: Classs extends Actor

- health: int

- bulletType: Enum

- grayMatter: int

- direction: float

- speed: float

- weaponsEnabled: boolean

- damageValue: int (How much base is damaged when tank is destroyed)

- x: int

- y: int

GameState: Class

(This object will be serialized sent back and forth from the server and client)

- team1: Team

- team2: Team

- turrets: libgdx.collections.Array<Turret>

- bullets: libgdx.collections.Array<Bullet>

- gameTime: int

Upgradable: Interface

- price: int

Bullet: Class extends Actor implements Upgradable

- damage: int

- speed: int

- sps: int (shots per second)

- color: Color

BaseRepairKit: Class implements Upgradable

healthIncrease: int (amount of health returned to the base)

Shield: Class implements Upgradable

- health: int (amount of shield remaining)

Turret: Class extends Actor implements Upgradable

- health: int

- bulletType: Bullet

- team: Team

Screens:

MainMenuScreen: Class implements gdx.Screen

- newGameButton: gdx.TextButton

- joinGameButton: gdx.TextButton

- statsButton: gdx.TextButton

- createViewProfileButton: gdx.TextButton

StatsScreen: Class implements gdx.Screen

- winLossRecordLabel: gdx.Label

- winLossRecord: int

- shotAccuracyLabel: gdx.Label

- shotAccuracy: double

- gamesPlayedLabel: gdx.Label

- gamesPlayed: int

- gamesWonLabel: gdx.Label

- gamesWon: int

- brainsDestroyedLabel: gdx.Label

- brainsDestroyed: int

- tanksDestroyedLabel: gdx.Label

- tanksDestroyed: int

- avgKillsPerLifeLabel: gdx.Label

- avgKillsPerLife: double

- grayMatterExtractedLabel: gdx.Label

- grayMatterExtracted: int

CreateProfileScreen: Class implements gdx.Screen

- nameLabel: gdx.Label

- nameTF: gdx.TextField

- passwordLabel: gdx.Label

- passwordTF: gdx.TextField

- createProfileButton: gdx.Button (validate name – must be unique)

CreateGameScreen extends gdx.Screen

- gameName: gdx.Label

- gameNameTF: gdx.TextField

- minNumPlayers: gdx.Label

- minNumPlayersTF: gdx.TextField

- startGameButton: gdx.ImageButton (validate game name – must be unique name)

WaitingScreen: gdx.Screen

- waitingLabel: gdx.Label

JoinGameScreen: gdx.Screen

- availableGames: gdx.Label

- gameName: gdx.Label (for however many available games there are)

- joinGameButton: gdx.Button (for number of available games)

- refreshGames() (update using Socket)

- waitingForGames: gdx.Label

BattleScreen: Class implements gdx.Screen

- battleScreen: BattleScreen

- utilityBar: UtilityBar

- gameStateBar: GameStateBar

- miniMap: MiniMap

- chatWindow: ChatWindow

- Render()

GameOverScreen: Class implements gdx.Screen

- teamLabel: gdx.Label

- kills: gdx.Label (int)

- points: gdx.Label (int)

- team 1: Table (using CellLabels - Kills and Points for each player on team)

- team 2: Table (using CellLabels - Kills and Points for each player on team)

- mainMenuButton: gdx.Button

- newGameButton: gdx.Button

ThinkTankServer: Class

- Games: static Map<int, Game>

- db: connection to the database

GameThread: Class extends Thread(class on server side that manages each game)

- team1: Team

- team2: Team

- ss: ServerSocket (used to receive messages from clients)

- void addPlayer(PlayerThread player, int team)

- boolean isReady()

- boolean isInterrupted()

- void run()

PlayerThread: Class extends Thread

- team: int

- client: Socket (used by server to send messages to client)

**Database:**

Engine: MySQL

ORM: Java JDBC

Tables:

   - Players:

* - winLossRecord: int
* - shotAccuracy: double
* - gamesPlayed: int
* - gamesWon: int
* - brainsDestroyed: int
* - tanksDestroyed: int
* - avgKillsPerLife: double
* - grayMatterExtracted: int
* - name: varchar(255, unique)
* - password: varchar (255)