

API Endpoints Documentation

Repository: 2048

File: README.md

```
# APCS - Final (2048)
By: Michael Borczuk, Khizer Shahid and Abir Taheer

## Steps for running the app

1. Download the repository either as a .zip file from GitHub or clone it with git
```commandline
git clone https://github.com/abir-taheer/2048.git
```

2. change into the code source folder
```commandline
cd src
```

3. Compile the Main.java class
```commandline
javac Main.java
```

4. Run the Main.class file now:
```commandline
java Main
```
```

File: Main.java

```
import info.gridworld.grid.UnboundedGrid;
import tiles.Tile;
import tiles.Tile2;

public class Main {

    public static void main(String[] args) {
        World world = new World();
        world.setup();
        world.show();
    }
}
```

File: World.java

```
/*
 * AP(r) Computer Science GridWorld Case Study:
 * Copyright(c) 2005-2006 Cay S. Horstmann (http://horstmann.com)
 *
 * This code is free software; you can redistribute it and/or modify
```

```

* it under the terms of the GNU General Public License as published by
* the Free Software Foundation.
*
* This code is distributed in the hope that it will be useful,
* but WITHOUT ANY WARRANTY; without even the implied warranty of
* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
* GNU General Public License for more details.
*
* @author Cay Horstmann
*/

```

```

package info.gridworld.world;

```

```

import info.gridworld.grid.BoundedGrid;
import info.gridworld.grid.Grid;
import info.gridworld.grid.Location;
import info.gridworld.gui.WorldFrame;
import java.util.ArrayList;
import java.util.Random;
import java.util.Set;
import java.util.TreeSet;
import javax.swing.*;

```

```

/**
 * A World is the mediator between a grid and the GridWorld GUI.
 * <br />
 * This class is not tested on the AP CS A and AB exams.
 */

```

```

public class World<T> {
    private static final int DEFAULT_ROWS = 4;
    private static final int DEFAULT_COLS = 4;
    private static final Random generator = new Random();
    private Grid<T> gr;
    private final Set<String> occupantClassNames;
    private final Set<String> gridClassNames;
    private String message;
    private JFrame frame;

```

```

    public World() {
        this(new BoundedGrid<T>(DEFAULT_ROWS, DEFAULT_COLS));
        message = null;
    }

```

```

    public World(Grid<T> g) {
        gr = g;
        gridClassNames = new TreeSet<String>();
        occupantClassNames = new TreeSet<String>();
        addGridClass("info.gridworld.grid.BoundedGrid");
        addGridClass("info.gridworld.grid.UnboundedGrid");
    }

```

```

/**
 * Constructs and shows a frame for this world.
 */

```

```

public void show() {
    if (frame == null) {
        frame = new WorldFrame<T>(this);
        frame.setVisible(true);
    } else frame.repaint();
}

public void showDialog(String message) {
    JOptionPane.showMessageDialog(frame, message);
}

public boolean confirmDialog(String title, String message) {
    return (
        JOptionPane.showConfirmDialog(
            frame,
            message,
            title,
            JOptionPane.YES_NO_OPTION
        ) ==
        0
    );
}

/**
 * Gets the grid managed by this world.
 *
 * @return the grid
 */
public Grid<T> getGrid() {
    return gr;
}

/**
 * Sets the grid managed by this world.
 *
 * @param newGrid the new grid
 */
public void setGrid(Grid<T> newGrid) {
    gr = newGrid;
    repaint();
}

/**
 * Gets the message to be displayed in the world frame above the grid.
 *
 * @return the message
 */
public String getMessage() {
    return message;
}

/**
 * Sets the message to be displayed in the world frame above the grid.
 *

```

```

* @param newMessage the new message
*/
public void setMessage(String newMessage) {
    message = newMessage;
    repaint();
}

/**
 * This method is called when the user clicks on the step button, or when
 * run mode has been activated by clicking the run button.
 */
public void step() {
    repaint();
}

/**
 * This method is called when the user clicks on a location in the
 * WorldFrame.
 *
 * @param loc the grid location that the user selected
 * @return true if the world consumes the click, or false if the GUI should
 * invoke the Location->Edit menu action
 */
public boolean locationClicked(Location loc) {
    return false;
}

/**
 * This method is called when a key was pressed. Override it if your world wants
 * to consume some keys (e.g. "1"-"9" for Sudoku). Don't consume plain arrow keys,
 * or the user loses the ability to move the selection square with the keyboard.
 *
 * @param description the string describing the key, in
 *
.
 \* @param loc the selected location in the grid at the time the key was pressed
 \* @return true if the world consumes the key press, false if the GUI should
 \* consume it.
 \*/
public boolean keyPressed\(String description, Location loc\) {
    return false;
}

/\*\*
 \* Gets a random empty location in this world.
 \*
 \* @return a random empty location
 \*/
public Location getRandomEmptyLocation\(\) {
    Grid<T> gr = getGrid\(\);
    int rows = gr.getNumRows\(\);
    int cols = gr.getNumCols\(\);

```

```

if (rows > 0 && cols > 0) { // bounded grid
// get all valid empty locations and pick one at random
ArrayList<Location> emptyLocs = new ArrayList<Location>();
for (int i = 0; i < rows; i++) for (int j = 0; j < cols; j++) {
Location loc = new Location(i, j);
if (gr.isValid(loc) && gr.get(loc) == null) emptyLocs.add(loc);
}
if (emptyLocs.size() == 0) return null;
int r = generator.nextInt(emptyLocs.size());
return emptyLocs.get(r);
} else { // unbounded grid
while (true) {
// keep generating a random location until an empty one is found
int r;
if (rows < 0) r =
(int) (DEFAULT_ROWS * generator.nextGaussian()); else r =
generator.nextInt(rows);
int c;
if (cols < 0) c =
(int) (DEFAULT_COLS * generator.nextGaussian()); else c =
generator.nextInt(cols);
Location loc = new Location(r, c);
if (gr.isValid(loc) && gr.get(loc) == null) return loc;
}
}
}

/**
 * Adds an occupant at a given location.
 *
 * @param loc      the location
 * @param occupant the occupant to add
 */
public void add(Location loc, T occupant) {
getGrid().put(loc, occupant);
repaint();
}

/**
 * Removes an occupant from a given location.
 *
 * @param loc the location
 * @return the removed occupant, or null if the location was empty
 */
public T remove(Location loc) {
T r = getGrid().remove(loc);
repaint();
return r;
}

/**
 * Adds a class to be shown in the "Set grid" menu.
 *
 * @param className the name of the grid class

```

```

*/
public void addGridClass(String className) {
    gridClassNames.add(className);
}

/**
 * Adds a class to be shown when clicking on an empty location.
 *
 * @param className the name of the occupant class
 */
public void addOccupantClass(String className) {
    occupantClassNames.add(className);
}

/**
 * Gets a set of grid classes that should be used by the world frame for
 * this world.
 *
 * @return the set of grid class names
 */
public Set<String> getGridClasses() {
    return gridClassNames;
}

/**
 * Gets a set of occupant classes that should be used by the world frame for
 * this world.
 *
 * @return the set of occupant class names
 */
public Set<String> getOccupantClasses() {
    return occupantClassNames;
}

private void repaint() {
    if (frame != null) frame.repaint();
}

/**
 * Returns a string that shows the positions of the grid occupants.
 */
public String toString() {
    String s = "";
    Grid<?> gr = getGrid();

    int rmin = 0;
    int rmax = gr.getNumRows() - 1;
    int cmin = 0;
    int cmax = gr.getNumCols() - 1;
    if (rmax < 0 || cmax < 0) { // unbounded grid
        for (Location loc : gr.getOccupiedLocations()) {
            int r = loc.getRow();
            int c = loc.getCol();
            if (r < rmin) rmin = r;

```

```

if (r > rmax) rmax = r;
if (c < cmin) cmin = c;
if (c > cmax) cmax = c;
}
}

for (int i = rmin; i <= rmax; i++) {
for (int j = cmin; j < cmax; j++) {
Object obj = gr.get(new Location(i, j));
if (obj == null) s += " "; else s +=
obj.toString().substring(0, 1);
}
s += "\n";
}
return s;
}
}

```

File: Actor.java

```

/*
 * AP(r) Computer Science GridWorld Case Study:
 * Copyright(c) 2005-2006 Cay S. Horstmann (http://horstmann.com)
 *
 * This code is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation.
 *
 * This code is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * @author Cay Horstmann
 */

package info.gridworld.actor;

import info.gridworld.grid.Grid;
import info.gridworld.grid.Location;
import java.awt.*;

/**
 * An Actor is an entity with a color and direction that can act.
 *   

 * The API of this class is testable on the AP CS A and AB exams.
 */
public class Actor {
    private Grid<Actor> grid;
    private Location location;
    private int direction;
    private Color color;

```



```

/**
 * Constructs a blue actor that is facing north.
 */
public Actor() {
    color = Color.BLUE;
    direction = Location.NORTH;
    grid = null;
    location = null;
}

/**
 * Gets the color of this actor.
 *
 * @return the color of this actor
 */
public Color getColor() {
    return color;
}

/**
 * Sets the color of this actor.
 *
 * @param newColor the new color
 */
public void setColor(Color newColor) {
    color = newColor;
}

/**
 * Gets the current direction of this actor.
 *
 * @return the direction of this actor, an angle between 0 and 359 degrees
 */
public int getDirection() {
    return direction;
}

/**
 * Sets the current direction of this actor.
 *
 * @param newDirection the new direction. The direction of this actor is set
 *                      to the angle between 0 and 359 degrees that is equivalent to
 *                      <code>newDirection</code>.
 */
public void setDirection(int newDirection) {
    direction = newDirection % Location.FULL_CIRCLE;
    if (direction < 0) direction += Location.FULL_CIRCLE;
}

/**
 * Gets the grid in which this actor is located.
 *
 * @return the grid of this actor, or <code>null</code> if this actor is

```

```

* not contained in a grid
*/
public Grid<Actor> getGrid() {
return grid;
}

/**
* Gets the location of this actor.
*
* @return the location of this actor, or <code>null</code> if this actor is
* not contained in a grid
*/
public Location getLocation() {
return location;
}

/**
* Puts this actor into a grid. If there is another actor at the given
* location, it is removed. <br />
* Precondition: (1) This actor is not contained in a grid (2)
* <code>loc</code> is valid in <code>gr</code>
*
* @param gr the grid into which this actor should be placed
* @param loc the location into which the actor should be placed
*/
public void putSelfInGrid(Grid<Actor> gr, Location loc) {
if (grid != null) throw new IllegalStateException(
"This actor is already contained in a grid."
);

Actor actor = gr.get(loc);
if (actor != null) actor.removeSelfFromGrid();
gr.put(loc, this);
grid = gr;
location = loc;
}

/**
* Removes this actor from its grid. <br />
* Precondition: This actor is contained in a grid
*/
public void removeSelfFromGrid() {
if (grid == null) throw new IllegalStateException(
"This actor is not contained in a grid."
);
if (grid.get(location) != this) throw new IllegalStateException(
"The grid contains a different actor at location " + location + "."
);

grid.remove(location);
grid = null;
location = null;
}

```

```

/**
 * Moves this actor to a new location. If there is another actor at the
 * given location, it is removed. <br />
 * Precondition: (1) This actor is contained in a grid (2)
 * <code>newLocation</code> is valid in the grid of this actor
 *
 * @param newLocation the new location
 */
public void moveTo(Location newLocation) {
    if (grid == null) throw new IllegalStateException(
        "This actor is not in a grid."
    );
    if (grid.get(location) != this) throw new IllegalStateException(
        "The grid contains a different actor at location " + location + "."
    );
    if (!grid.isValid(newLocation)) throw new IllegalArgumentException(
        "Location " + newLocation + " is not valid."
    );

    if (newLocation.equals(location)) return;
    grid.remove(location);
    Actor other = grid.get(newLocation);
    if (other != null) other.removeSelfFromGrid();
    location = newLocation;
    grid.put(location, this);
}

/**
 * Reverses the direction of this actor. Override this method in subclasses
 * of <code>Actor</code> to define types of actors with different behavior
 */
public void act() {
    setDirection(getDirection() + Location.HALF_CIRCLE);
}

/**
 * Creates a string that describes this actor.
 *
 * @return a string with the location, direction, and color of this actor
 */
public String toString() {
    return (
        getClass().getName() +
        "[location=" +
        location +
        ",direction=" +
        direction +
        ",color=" +
        color +
        "]"
    );
}
}

```

File: ActorWorld.java

```
/*
 * AP(r) Computer Science GridWorld Case Study:
 * Copyright(c) 2005-2006 Cay S. Horstmann (http://horstmann.com)
 *
 * This code is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation.
 *
 * This code is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * @author Cay Horstmann
 */

package info.gridworld.actor;

import info.gridworld.grid.Grid;
import info.gridworld.grid.Location;
import info.gridworld.world.World;
import java.util.ArrayList;

/**
 * An ActorWorld is occupied by actors. <br />
 * This class is not tested on the AP CS A and AB exams.
 */

public class ActorWorld extends World<Actor> {
    private static final String DEFAULT_MESSAGE =
        "Use the arrow keys on your keyboard to play.";

    /**
     * Constructs an actor world with a default grid.
     */
    public ActorWorld() {}

    /**
     * Constructs an actor world with a given grid.
     *
     * @param grid the grid for this world.
     */
    public ActorWorld(Grid<Actor> grid) {
        super(grid);
    }

    public void show() {
        if (getMessage() == null) setMessage(DEFAULT_MESSAGE);
        super.show();
    }
}
```

```

public void step() {
    Grid<Actor> gr = getGrid();
    ArrayList<Actor> actors = new ArrayList<Actor>();
    for (Location loc : gr.getOccupiedLocations()) actors.add(gr.get(loc));

    for (Actor a : actors) {
        // only act if another actor hasn't removed a
        if (a.getGrid() == gr) a.act();
    }
}

/**
 * Adds an actor to this world at a given location.
 *
 * @param loc        the location at which to add the actor
 * @param occupant the actor to add
 */
public void add(Location loc, Actor occupant) {
    occupant.putSelfInGrid(getGrid(), loc);
}

/**
 * Adds an occupant at a random empty location.
 *
 * @param occupant the occupant to add
 */
public void add(Actor occupant) {
    Location loc = getRandomEmptyLocation();
    if (loc != null) add(loc, occupant);
}

/**
 * Removes an actor from this world.
 *
 * @param loc the location from which to remove an actor
 * @return the removed actor, or null if there was no actor at the given
 *         location.
 */
public Actor remove(Location loc) {
    Actor occupant = getGrid().get(loc);
    if (occupant == null) return null;
    occupant.removeSelfFromGrid();
    return occupant;
}
}

```

File: AbstractGrid.java

```

/*
 * AP(r) Computer Science GridWorld Case Study:
 * Copyright(c) 2005-2006 Cay S. Horstmann (http://horstmann.com)

```

```

*
* This code is free software; you can redistribute it and/or modify
* it under the terms of the GNU General Public License as published by
* the Free Software Foundation.
*
* This code is distributed in the hope that it will be useful,
* but WITHOUT ANY WARRANTY; without even the implied warranty of
* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
* GNU General Public License for more details.
*
* @author Cay Horstmann
*/

package info.gridworld.grid;

import java.util.ArrayList;

/**
 * AbstractGrid contains the methods that are common to grid
 * implementations. <br />
 * The implementation of this class is testable on the AP CS AB exam.
 */
public abstract class AbstractGrid<E> implements Grid<E> {

    public ArrayList<E> getNeighbors(Location loc) {
        ArrayList<E> neighbors = new ArrayList<E>();
        for (Location neighborLoc : getOccupiedAdjacentLocations(
            loc
        )) neighbors.add(get(neighborLoc));
        return neighbors;
    }

    public ArrayList<Location> getValidAdjacentLocations(Location loc) {
        ArrayList<Location> locs = new ArrayList<Location>();

        int d = Location.NORTH;
        for (int i = 0; i < Location.FULL_CIRCLE / Location.HALF_RIGHT; i++) {
            Location neighborLoc = loc.getAdjacentLocation(d);
            if (isValid(neighborLoc)) locs.add(neighborLoc);
            d = d + Location.HALF_RIGHT;
        }
        return locs;
    }

    public ArrayList<Location> getEmptyAdjacentLocations(Location loc) {
        ArrayList<Location> locs = new ArrayList<Location>();
        for (Location neighborLoc : getValidAdjacentLocations(loc)) {
            if (get(neighborLoc) == null) locs.add(neighborLoc);
        }
        return locs;
    }

    public ArrayList<Location> getOccupiedAdjacentLocations(Location loc) {
        ArrayList<Location> locs = new ArrayList<Location>();

```

```

for (Location neighborLoc : getValidAdjacentLocations(loc)) {
    if (get(neighborLoc) != null) locs.add(neighborLoc);
}
return locs;
}

/**
 * Creates a string that describes this grid.
 *
 * @return a string with descriptions of all objects in this grid (not
 * necessarily in any particular order), in the format {loc=obj, loc=obj,
 * ...}
 */
public String toString() {
    String s = "{";
    for (Location loc : getOccupiedLocations()) {
        if (s.length() > 1) s += ", ";
        s += loc + "=" + get(loc);
    }
    return s + "}";
}
}

```

File: BoundedGrid.java

```

/**
 * AP(r) Computer Science GridWorld Case Study:
 * Copyright(c) 2002-2006 College Entrance Examination Board
 * (http://www.collegeboard.com).
 *
 * This code is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation.
 *
 * This code is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * @author Alyce Brady
 * @author APCS Development Committee
 * @author Cay Horstmann
 */

package info.gridworld.grid;

import java.util.ArrayList;

/**
 * A BoundedGrid is a rectangular grid with a finite number of
 * rows and columns. <br />
 * The implementation of this class is testable on the AP CS AB exam.

```

```

*/
public class BoundedGrid<E> extends AbstractGrid<E> {
private final Object[][] occupantArray; // the array storing the grid elements

/**
 * Constructs an empty bounded grid with the given dimensions.
 * (Precondition: <code>rows > 0</code> and <code>cols > 0</code>.)
 *
 * @param rows number of rows in BoundedGrid
 * @param cols number of columns in BoundedGrid
 */
public BoundedGrid(int rows, int cols) {
if (rows <= 0) throw new IllegalArgumentException("rows <= 0");
if (cols <= 0) throw new IllegalArgumentException("cols <= 0");
occupantArray = new Object[rows][cols];
}

public int getNumRows() {
return occupantArray.length;
}

public int getNumCols() {
// Note: according to the constructor precondition, numRows() > 0, so
// theGrid[0] is non-null.
return occupantArray[0].length;
}

public boolean isValid(Location loc) {
return (
0 <= loc.getRow() &&
loc.getRow() < getNumRows() &&
0 <= loc.getCol() &&
loc.getCol() < getNumCols()
);
}

public ArrayList<Location> getOccupiedLocations() {
ArrayList<Location> theLocations = new ArrayList<Location>();

// Look at all grid locations.
for (int r = 0; r < getNumRows(); r++) {
for (int c = 0; c < getNumCols(); c++) {
// If there's an object at this location, put it in the array.
Location loc = new Location(r, c);
if (get(loc) != null) theLocations.add(loc);
}
}

return theLocations;
}

public E get(Location loc) {
if (!isValid(loc)) throw new IllegalArgumentException(
"Location " + loc + " is not valid"

```



```

    );
    return (E) occupantArray[loc.getRow()][loc.getCol()]; // unavoidable warning
}

public E put(Location loc, E obj) {
    if (!isValid(loc)) throw new IllegalArgumentException(
        "Location " + loc + " is not valid"
    );
    if (obj == null) throw new NullPointerException("obj == null");

    // Add the object to the grid.
    E oldOccupant = get(loc);
    occupantArray[loc.getRow()][loc.getCol()] = obj;
    return oldOccupant;
}

public E remove(Location loc) {
    if (!isValid(loc)) throw new IllegalArgumentException(
        "Location " + loc + " is not valid"
    );

    // Remove the object from the grid.
    E r = get(loc);
    occupantArray[loc.getRow()][loc.getCol()] = null;
    return r;
}
}

```

File: Grid.java

```

/*
 * AP(r) Computer Science GridWorld Case Study:
 * Copyright(c) 2002-2006 College Entrance Examination Board
 * (http://www.collegeboard.com).
 *
 * This code is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation.
 *
 * This code is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * @author Alyce Brady
 * @author APCS Development Committee
 * @author Cay Horstmann
 */

package info.gridworld.grid;

import java.util.ArrayList;

```

```

/**
 * <code>Grid</code> provides an interface for a two-dimensional, grid-like
 * environment containing arbitrary objects. <br />
 * This interface is testable on the AP CS A and AB exams.
 */
public interface Grid<E> {
    /**
     * Returns the number of rows in this grid.
     *
     * @return the number of rows, or -1 if this grid is unbounded
     */
    int getNumRows();

    /**
     * Returns the number of columns in this grid.
     *
     * @return the number of columns, or -1 if this grid is unbounded
     */
    int getNumCols();

    /**
     * Checks whether a location is valid in this grid. <br />
     * Precondition: <code>loc</code> is not <code>null</code>
     *
     * @param loc the location to check
     * @return <code>true</code> if <code>loc</code> is valid in this grid,
     * <code>false</code> otherwise
     */
    boolean isValid(Location loc);

    /**
     * Puts an object at a given location in this grid. <br />
     * Precondition: (1) <code>loc</code> is valid in this grid (2)
     * <code>obj</code> is not <code>null</code>
     *
     * @param loc the location at which to put the object
     * @param obj the new object to be added
     * @return the object previously at <code>loc</code> (or <code>null</code>
     * if the location was previously unoccupied)
     */
    E put(Location loc, E obj);

    /**
     * Removes the object at a given location from this grid. <br />
     * Precondition: <code>loc</code> is valid in this grid
     *
     * @param loc the location of the object that is to be removed
     * @return the object that was removed (or <code>null</code> if the location
     * is unoccupied)
     */
    E remove(Location loc);

    /**

```

```

* Returns the object at a given location in this grid. <br />
* Precondition: <code>loc</code> is valid in this grid
*
* @param loc a location in this grid
* @return the object at location <code>loc</code> (or <code>null</code>
* if the location is unoccupied)
*/
E get(Location loc);

/**
* Gets the locations in this grid that contain objects.
*
* @return an array list of all occupied locations in this grid
*/
ArrayList<Location> getOccupiedLocations();

/**
* Gets the valid locations adjacent to a given location in all eight
* compass directions (north, northeast, east, southeast, south, southwest,
* west, and northwest). <br />
* Precondition: <code>loc</code> is valid in this grid
*
* @param loc a location in this grid
* @return an array list of the valid locations adjacent to <code>loc</code>
* in this grid
*/
ArrayList<Location> getValidAdjacentLocations(Location loc);

/**
* Gets the valid empty locations adjacent to a given location in all eight
* compass directions (north, northeast, east, southeast, south, southwest,
* west, and northwest). <br />
* Precondition: <code>loc</code> is valid in this grid
*
* @param loc a location in this grid
* @return an array list of the valid empty locations adjacent to
* <code>loc</code> in this grid
*/
ArrayList<Location> getEmptyAdjacentLocations(Location loc);

/**
* Gets the valid occupied locations adjacent to a given location in all
* eight compass directions (north, northeast, east, southeast, south,
* southwest, west, and northwest). <br />
* Precondition: <code>loc</code> is valid in this grid
*
* @param loc a location in this grid
* @return an array list of the valid occupied locations adjacent to
* <code>loc</code> in this grid
*/
ArrayList<Location> getOccupiedAdjacentLocations(Location loc);

/**
* Gets the neighboring occupants in all eight compass directions (north,

```

```

* northeast, east, southeast, south, southwest, west, and northwest).
* <br />
* Precondition: <code>loc</code> is valid in this grid
*
* @param loc a location in this grid
* @return returns an array list of the objects in the occupied locations
* adjacent to <code>loc</code> in this grid
*/
ArrayList<E> getNeighbors(Location loc);
}

```

File: Location.java

```

/*
* AP(r) Computer Science GridWorld Case Study:
* Copyright(c) 2002-2006 College Entrance Examination Board
* (http://www.collegeboard.com).
*
* This code is free software; you can redistribute it and/or modify
* it under the terms of the GNU General Public License as published by
* the Free Software Foundation.
*
* This code is distributed in the hope that it will be useful,
* but WITHOUT ANY WARRANTY; without even the implied warranty of
* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
* GNU General Public License for more details.
*
* @author Alyce Brady
* @author Chris Nevison
* @author APCS Development Committee
* @author Cay Horstmann
*/

package info.gridworld.grid;

/**
* A <code>Location</code> object represents the row and column of a location
* in a two-dimensional grid. <br />
* The API of this class is testable on the AP CS A and AB exams.
*/
public class Location implements Comparable {
/**
* The turn angle for turning 90 degrees to the left.
*/
public static final int LEFT = -90;
/**
* The turn angle for turning 90 degrees to the right.
*/
public static final int RIGHT = 90;
/**
* The turn angle for turning 45 degrees to the left.
*/

```

```

public static final int HALF_LEFT = -45;
/**
 * The turn angle for turning 45 degrees to the right.
 */
public static final int HALF_RIGHT = 45;
/**
 * The turn angle for turning a full circle.
 */
public static final int FULL_CIRCLE = 360;
/**
 * The turn angle for turning a half circle.
 */
public static final int HALF_CIRCLE = 180;
/**
 * The turn angle for making no turn.
 */
public static final int AHEAD = 0;
/**
 * The compass direction for north.
 */
public static final int NORTH = 0;
/**
 * The compass direction for northeast.
 */
public static final int NORTHEAST = 45;
/**
 * The compass direction for east.
 */
public static final int EAST = 90;
/**
 * The compass direction for southeast.
 */
public static final int SOUTHEAST = 135;
/**
 * The compass direction for south.
 */
public static final int SOUTH = 180;
/**
 * The compass direction for southwest.
 */
public static final int SOUTHWEST = 225;
/**
 * The compass direction for west.
 */
public static final int WEST = 270;
/**
 * The compass direction for northwest.
 */
public static final int NORTHWEST = 315;
private final int row; // row location in grid
private final int col; // column location in grid

/**
 * Constructs a location with given row and column coordinates.

```

```

*
* @param r the row
* @param c the column
*/
public Location(int r, int c) {
    row = r;
    col = c;
}

/**
 * Gets the row coordinate.
 *
 * @return the row of this location
 */
public int getRow() {
    return row;
}

/**
 * Gets the column coordinate.
 *
 * @return the column of this location
 */
public int getCol() {
    return col;
}

/**
 * Gets the adjacent location in any one of the eight compass directions.
 *
 * @param direction the direction in which to find a neighbor location
 * @return the adjacent location in the direction that is closest to
 * <tt>direction</tt>
 */
public Location getAdjacentLocation(int direction) {
    // reduce mod 360 and round to closest multiple of 45
    int adjustedDirection = (direction + HALF_RIGHT / 2) % FULL_CIRCLE;
    if (adjustedDirection < 0) adjustedDirection += FULL_CIRCLE;

    adjustedDirection = (adjustedDirection / HALF_RIGHT) * HALF_RIGHT;
    int dc = 0;
    int dr = 0;
    if (adjustedDirection == EAST) dc = 1; else if (
    adjustedDirection == SOUTHEAST
    ) {
        dc = 1;
        dr = 1;
    } else if (adjustedDirection == SOUTH) dr = 1; else if (
    adjustedDirection == SOUTHWEST
    ) {
        dc = -1;
        dr = 1;
    } else if (adjustedDirection == WEST) dc = -1; else if (
    adjustedDirection == NORTHWEST

```

```

    ) {
    dc = -1;
    dr = -1;
    } else if (adjustedDirection == NORTH) dr = -1; else if (
adjustedDirection == NORTHEAST
    ) {
    dc = 1;
    dr = -1;
    }
    return new Location(getRow() + dr, getCol() + dc);
    }

/**
 * Returns the direction from this location toward another location. The
 * direction is rounded to the nearest compass direction.
 *
 * @param target a location that is different from this location
 * @return the closest compass direction from this location toward
 * <code>target</code>
 */
public int getDirectionToward(Location target) {
    int dx = target.getCol() - getCol();
    int dy = target.getRow() - getRow();
    // y axis points opposite to mathematical orientation
    int angle = (int) Math.toDegrees(Math.atan2(-dy, dx));

    // mathematical angle is counterclockwise from x-axis,
    // compass angle is clockwise from y-axis
    int compassAngle = RIGHT - angle;
    // prepare for truncating division by 45 degrees
    compassAngle += HALF_RIGHT / 2;
    // wrap negative angles
    if (compassAngle < 0) compassAngle += FULL_CIRCLE;
    // round to nearest multiple of 45
    return (compassAngle / HALF_RIGHT) * HALF_RIGHT;
}

/**
 * Indicates whether some other <code>Location</code> object is "equal to"
 * this one.
 *
 * @param other the other location to test
 * @return <code>true</code> if <code>other</code> is a
 * <code>Location</code> with the same row and column as this location;
 * <code>false</code> otherwise
 */
public boolean equals(Object other) {
    if (!(other instanceof Location)) return false;

    Location otherLoc = (Location) other;
    return getRow() == otherLoc.getRow() && getCol() == otherLoc.getCol();
}

/**

```

```

* Generates a hash code.
*
* @return a hash code for this location
*/
public int hashCode() {
return getRow() * 3737 + getCol();
}

/**
* Compares this location to <code>other</code> for ordering. Returns a
* negative integer, zero, or a positive integer as this location is less
* than, equal to, or greater than <code>other</code>. Locations are
* ordered in row-major order. <br />
* (Precondition: <code>other</code> is a <code>Location</code> object.)
*
* @param other the other location to test
* @return a negative integer if this location is less than
* <code>other</code>, zero if the two locations are equal, or a positive
* integer if this location is greater than <code>other</code>
*/
public int compareTo(Object other) {
Location otherLoc = (Location) other;
if (getRow() < otherLoc.getRow()) return -1;
if (getRow() > otherLoc.getRow()) return 1;
if (getCol() < otherLoc.getCol()) return -1;
if (getCol() > otherLoc.getCol()) return 1;
return 0;
}

/**
* Creates a string that describes this location.
*
* @return a string with the row and column of this location, in the format
* (row, col)
*/
public String toString() {
return "(" + getRow() + ", " + getCol() + ")";
}
}

```

File: AbstractDisplay.java

```

/*
* AP(r) Computer Science GridWorld Case Study:
* Copyright(c) 2002-2006 College Entrance Examination Board
* (http://www.collegeboard.com).
*
* This code is free software; you can redistribute it and/or modify
* it under the terms of the GNU General Public License as published by
* the Free Software Foundation.
*
* This code is distributed in the hope that it will be useful,

```



```

* but WITHOUT ANY WARRANTY; without even the implied warranty of
* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
* GNU General Public License for more details.
*
* @author Julie Zelenski
* @author Cay Horstmann
*/

```

```

package info.gridworld.gui;

```

```

import java.awt.*;
import java.beans.BeanInfo;
import java.beans.Introspector;
import java.beans.PropertyDescriptor;
import java.lang.reflect.Method;

```

```

/**
 * This class provides common implementation code for drawing objects. It will
 * translate, scale, and rotate the graphics system as needed and then invoke
 * its abstract <code>draw</code> method. Subclasses of this abstract class
 * define <code>draw</code> to display an object in a fixed size and
 * orientation. <br />
 * This code is not tested on the AP CS A and AB exams. It contains GUI
 * implementation details that are not intended to be understood by AP CS
 * students.
 */

```

```

public abstract class AbstractDisplay implements Display {

```

```

    public static Object getProperty(Object obj, String propertyName) {
        if (obj == null) return null;
        try {
            BeanInfo info = Introspector.getBeanInfo(obj.getClass());
            PropertyDescriptor[] descriptors = info.getPropertyDescriptors();
            for (int i = 0; i < descriptors.length; i++) {
                if (descriptors[i].getName().equals(propertyName)) {
                    Method getter = descriptors[i].getReadMethod();
                    if (getter == null) return null;
                    try {
                        return getter.invoke(obj);
                    } catch (Exception ex) {
                        System.out.println(descriptors[i].getName());
                        return null;
                    }
                }
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
        return null;
    }

```

```

/**
 * Draw the given object. Subclasses should implement this method to draw

```

```

* the occupant facing North in a cell of size (1,1) centered around (0,0) on
* the drawing surface. (All scaling/rotating has been done beforehand).
*
* @param obj the occupant we want to draw
* @param comp the component on which to draw
* @param g2 the graphics context
*/
public abstract void draw(Object obj, Component comp, Graphics2D g2);

/**
* Draw the given object. Scales and rotates the coordinate appropriately
* then invokes the simple draw method above that is only responsible for
* drawing a unit-length occupant facing North.
*
* @param obj the occupant we want to draw
* @param comp the component on which to draw
* @param g2 the graphics context
* @param rect rectangle in which to draw
*/
public void draw(
Object obj,
Component comp,
Graphics2D g2,
Rectangle rect
) {
float scaleFactor = Math.min(rect.width, rect.height);
g2 = (Graphics2D) g2.create();

// Translate to center of the object
g2.translate(rect.x + rect.width / 2.0, rect.y + rect.height / 2.0);

// Rotate drawing surface before drawing to capture object's
// orientation (direction).
if (obj != null) {
Integer direction = (Integer) getProperty(obj, "direction");
int rotationInDegrees = direction == null
? 0
: direction.intValue();
g2.rotate(Math.toRadians(rotationInDegrees));
}
// Scale to size of rectangle, adjust stroke back to 1-pixel wide
g2.scale(scaleFactor, scaleFactor);
g2.setStroke(new BasicStroke(1.0f / scaleFactor));
draw(obj, comp, g2);
}
}

```

File: ColorEditor.java

```

/*
* AP(r) Computer Science GridWorld Case Study:
* Copyright(c) 2005-2006 Cay S. Horstmann (http://horstmann.com)

```

```

*
* This code is free software; you can redistribute it and/or modify
* it under the terms of the GNU General Public License as published by
* the Free Software Foundation.
*
* This code is distributed in the hope that it will be useful,
* but WITHOUT ANY WARRANTY; without even the implied warranty of
* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
* GNU General Public License for more details.
*
* @author Cay Horstmann
*/

```

```
package info.gridworld.gui;
```

```
import java.awt.*;
import java.beans.PropertyEditorSupport;
import javax.swing.*;
```

```
/**
 * A property editor for the Color type. <br />
 * This code is not tested on the AP CS A and AB exams. It contains GUI
 * implementation details that are not intended to be understood by AP CS
 * students.
 */

```

```
public class ColorEditor extends PropertyEditorSupport {
    private static final Color[] colorValues = {
        Color.BLACK,
        Color.BLUE,
        Color.CYAN,
        Color.DARK_GRAY,
        Color.GRAY,
        Color.GREEN,
        Color.LIGHT_GRAY,
        Color.MAGENTA,
        Color.ORANGE,
        Color.PINK,
        Color.RED,
        Color.WHITE,
        Color.YELLOW,
    };
    private static final ColorIcon[] colorIcons;

    static {
        colorIcons = new ColorIcon[colorValues.length + 1];
        colorIcons[0] = new RandomColorIcon();
        for (int i = 0; i < colorValues.length; i++) colorIcons[i + 1] =
            new SolidColorIcon(colorValues[i]);
    }

```

```
private final JComboBox combo;
```

```
public ColorEditor() {
    combo = new JComboBox(colorIcons);

```

```

}

public Object getValue() {
    ColorIcon value = (ColorIcon) combo.getSelectedItem();
    return value.getColor();
}

public boolean supportsCustomEditor() {
    return true;
}

public Component getCustomEditor() {
    combo.setSelectedItem(0);
    return combo;
}

private interface ColorIcon extends Icon {
    int WIDTH = 120;
    int HEIGHT = 20;

    Color getColor();
}

private static class SolidColorIcon implements ColorIcon {
    private final Color color;

    public SolidColorIcon(Color c) {
        color = c;
    }

    public Color getColor() {
        return color;
    }

    public int getIconWidth() {
        return WIDTH;
    }

    public int getIconHeight() {
        return HEIGHT;
    }

    public void paintIcon(Component c, Graphics g, int x, int y) {
        Rectangle r = new Rectangle(x, y, WIDTH - 1, HEIGHT - 1);
        Graphics2D g2 = (Graphics2D) g;
        Color oldColor = g2.getColor();
        g2.setColor(color);
        g2.fill(r);
        g2.setColor(Color.BLACK);
        g2.draw(r);
        g2.setColor(oldColor);
    }
}

```

```

private static class RandomColorIcon implements ColorIcon {

public Color getColor() {
return new Color((int) (Math.random() * 256 * 256 * 256));
}

public int getIconWidth() {
return WIDTH;
}

public int getIconHeight() {
return HEIGHT;
}

public void paintIcon(Component c, Graphics g, int x, int y) {
Rectangle r = new Rectangle(x, y, WIDTH - 1, HEIGHT - 1);
Graphics2D g2 = (Graphics2D) g;
Color oldColor = g2.getColor();
Rectangle r1 = new Rectangle(x, y, WIDTH / 4, HEIGHT - 1);
for (int i = 0; i < 4; i++) {
g2.setColor(getColor());
g2.fill(r1);
r1.translate(WIDTH / 4, 0);
}
g2.setColor(Color.BLACK);
g2.draw(r);
g2.setColor(oldColor);
}
}
}

```

File: DefaultDisplay.java

```

/*
 * AP(r) Computer Science GridWorld Case Study:
 * Copyright(c) 2002-2006 College Entrance Examination Board
 * (http://www.collegeboard.com).
 *
 * This code is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation.
 *
 * This code is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * @author Alyce Brady
 * @author Jeff Raab, Northeastern University
 * @author Cay Horstmann
 */

```

```
package info.gridworld.gui;

import java.awt.*;
import java.awt.font.FontRenderContext;
import java.awt.font.LineMetrics;
import java.awt.geom.Rectangle2D;

/**
 * The DefaultDisplay draws the object's text property with a background color
 * given by the object's color property. <br />
 * This code is not tested on the AP CS A and AB exams. It contains GUI
 * implementation details that are not intended to be understood by AP CS
 * students.
 */
public class DefaultDisplay implements Display {
    private static final int MAX_TEXT_LENGTH = 8;

    /**
     * Draw the given object. This implementation draws a string with
     * a background color. The background color is the value
     * of the color property, or, if there is no such property
     * and the object is an instance of Color, the object itself.
     * The string is the text property, or if there is no such
     * property, the result of calling toString. The string
     * is clipped to 8 characters.
     *
     * @param obj    object we want to draw
     * @param comp   component on which to draw
     * @param g2     drawing surface
     * @param rect   rectangle in which to draw
     */
    public void draw(
        Object obj,
        Component comp,
        Graphics2D g2,
        Rectangle rect
    ) {
        Color color = (Color) AbstractDisplay.getProperty(obj, "color");
        if (color == null && obj instanceof Color) color = (Color) obj;
        Color textColor = (Color) AbstractDisplay.getProperty(obj, "textColor");
        if (textColor == null) textColor = Color.BLACK;
        if (color != null) {
            g2.setPaint(color);
            g2.fill(rect);

            if (color.equals(textColor)) {
                textColor =
                    new Color(
                        255 - textColor.getRed(),
                        255 - textColor.getGreen(),
                        255 - textColor.getBlue()
                    );
            }
        }
    }
}
```

```

String text = (String) AbstractDisplay.getProperty(obj, "text");
if (text == null && !(obj instanceof Color)) {
text = "" + obj;
}
if (text == null) return;
if (text.length() > MAX_TEXT_LENGTH) text =
text.substring(0, MAX_TEXT_LENGTH) + "...";
paintCenteredText(g2, text, rect, 2.5, textColor);
}

/**
 * Paint a horizontally and vertically-centered text string.
 *
 * @param g2          drawing surface
 * @param s           string to draw (centered)
 * @param rect        the bounding rectangle
 * @param fontHeight  the desired height of the font. (The font will be
 *                    shrunk in increments of sqrt(2)/2 if the text is too large.)
 * @param color       the color in which to draw the text
 */
protected void paintCenteredText(
Graphics2D g2,
String s,
Rectangle rect,
double fontHeight,
Color color
) {
g2 = (Graphics2D) g2.create();
g2.setRenderingHint(
RenderingHints.KEY_ANTIALIASING,
RenderingHints.VALUE_ANTIALIAS_ON
);
g2.setPaint(color);
Rectangle2D bounds = null;
LineMetrics lm = null;
boolean done = false;
// shrink font in increments of sqrt(2)/2 until string fits
while (!done) {
g2.setFont(
new Font(
"SansSerif",
Font.BOLD,
(int) (fontHeight * rect.height)
)
);
FontRenderContext frc = g2.getFontRenderContext();
bounds = g2.getFont().getStringBounds(s, frc);
if (bounds.getWidth() > rect.getWidth()) fontHeight =
fontHeight * Math.sqrt(2) / 2; else {
done = true;
lm = g2.getFont().getLineMetrics(s, frc);
}
}
float centerX = rect.x + rect.width / 2;

```

```

float centerY = rect.y + rect.height / 2;
float leftX = centerX - (float) bounds.getWidth() / 2;
float baselineY = centerY - lm.getHeight() / 2 + lm.getAscent();
g2.drawString(s, leftX, baselineY);
g2.dispose();
}
}

```

File: Display.java

```

/*
 * AP(r) Computer Science GridWorld Case Study:
 * Copyright(c) 2002-2006 College Entrance Examination Board
 * (http://www.collegeboard.com).
 *
 * This code is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation.
 *
 * This code is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * @author Alyce Brady
 * @author Cay Horstmann
 */

package info.gridworld.gui;

import java.awt.*;

/**
 * The <code>Display</code> interface contains the method needed to display a
 * grid object. <br />
 * This code is not tested on the AP CS A and AB exams. It contains GUI
 * implementation details that are not intended to be understood by AP CS
 * students.
 */
public interface Display {
    /**
     * Method invoked to draw an object.
     *
     * @param obj    object we want to draw
     * @param comp   component on which to draw
     * @param g2     drawing surface
     * @param rect   rectangle in which to draw
     */
    void draw(Object obj, Component c, Graphics2D g2, Rectangle rect);
}

```


File: DisplayMap.java

```
/*
 * AP(r) Computer Science GridWorld Case Study:
 * Copyright(c) 2002-2006 College Entrance Examination Board
 * (http://www.collegeboard.com).
 *
 * This code is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation.
 *
 * This code is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * @author Alyce Brady
 * @author Jeff Raab, Northeastern University
 * @author Cay Horstmann
 */

package info.gridworld.gui;

import java.awt.*;
import java.awt.geom.AffineTransform;
import java.util.HashMap;
import javax.swing.*;

/**
 * <code>DisplayMap</code> is a collection that maps grid occupant
 * classes to objects that know how to display them. <br />
 * This code is not tested on the AP CS A and AB exams. It contains GUI
 * implementation details that are not intended to be understood by AP CS
 * students.
 */
public class DisplayMap {
    private final HashMap<Class, Display> map = new HashMap<Class, Display>();
    private final Display defaultDisplay = new DefaultDisplay();

    /**
     * Associates a display object with a grid occupant class.
     *
     * @param the occupant class
     * @return the ImageDisplay or (classname)Display object to display it,
     * or null if none was found
     */

    private Display createDisplay(Class cl) {
        try {
            String className = cl.getName();
            Class dcl = Class.forName(className + "Display");
            if (Display.class.isAssignableFrom(dcl)) {
```

```

Display display = (Display) dcl.newInstance();
map.put(cl, display);
return display;
}
} catch (Exception e) {
// oh well...
}

try {
ImageDisplay display = new ImageDisplay(cl);
map.put(cl, display);
return display;
} catch (Exception e) {
// oh well...
}

return null;
}

/**
 * Finds a display class that knows how to display the given object.
 *
 * @param obj the object to display
 */
public Display findDisplayFor(Class cl) {
// Go up through the class hierarchy for obj and see
// if there is a display for its class or superclasses.

if (cl == Object.class) return defaultDisplay;
Display display = map.get(cl);
if (display != null) return display;
display = createDisplay(cl);
if (display != null) {
map.put(cl, display);
return display;
}
display = findDisplayFor(cl.getSuperclass());
map.put(cl, display);
return display;
}

/**
 * Gets an icon to display a class in a menu
 *
 * @param cl the class
 * @param w the icon width
 * @param h the icon height
 * @return the icon
 */
public Icon getIcon(Class cl, int w, int h) {
return new DisplayIcon(cl, w, h);
}

private class DisplayIcon implements Icon {

```

```

private final Display displayObj;
private final int width;
private final int height;

public DisplayIcon(Class cl, int w, int h) {
displayObj = findDisplayFor(cl);
width = w;
height = h;
}

public int getIconWidth() {
return width;
}

public int getIconHeight() {
return height;
}

public void paintIcon(Component comp, Graphics g, int x, int y) {
Graphics2D g2 = (Graphics2D) g;
AffineTransform savedTransform = g2.getTransform(); // save current
displayObj.draw(
null,
comp,
g2,
new Rectangle(x, y, getIconWidth(), getIconHeight())
);
g2.setTransform(savedTransform); // restore coordinate system
}
}
}

```

File: Tile.java

```

package tiles;

import info.gridworld.actor.Actor;

public class Tile extends Actor {
protected int originalValue = 0;
protected int value;

public Tile() {
super();
}

public Tile getRefreshedTile() {
if (originalValue == value) {
return this;
}

Tile newTile = new Tile();

```

```
if (value == 2) {
newTile = new Tile2();
}

if (value == 4) {
newTile = new Tile4();
}

if (value == 8) {
newTile = new Tile8();
}

if (value == 16) {
newTile = new Tile16();
}

if (value == 32) {
newTile = new Tile32();
}

if (value == 64) {
newTile = new Tile64();
}

if (value == 128) {
newTile = new Tile128();
}

if (value == 256) {
newTile = new Tile256();
}

if (value == 512) {
newTile = new Tile512();
}

if (value == 1024) {
newTile = new Tile1024();
}

if (value == 2048) {
newTile = new Tile2048();
}

return newTile;
}

public int getValue() {
return value;
}

public void setValue(int value) {
this.value = value;
}
```

```
}
```

```
public boolean willChange() {  
    return originalValue != value;  
}  
}
```

File: Tile1024.java

```
package tiles;
```

```
public class Tile1024 extends Tile {
```

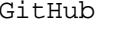
```
    public Tile1024() {  
        this.originalValue = 1024;  
        this.value = originalValue;  
    }  
}
```

Repository: abir-taheer

File: README.md

Hi there ?

I'm Abir and I do full-stack! You'll also find me taking [photos](https://abir.photos) from time to time :)

![My  GitHub stats](https://github-readme-stats.vercel.app/api?username=abir-taheer&count_private=true)

Repository: abir.nyc

File: README.md

abir.nyc

Personal site

This is a [Next.js](https://nextjs.org/) project bootstrapped with [create-next-app](https://github.com/vercel/next.js/tree/canary/packages/create-next-app).

Getting Started

First, run the development server:

```
```bash
npm run dev
or
yarn dev
```
```

Open http://localhost:3000 with your browser to see the result.

You can start editing the page by modifying `pages/index.js`. The page auto-updates as you edit the file.

[API routes](https://nextjs.org/docs/api-routes/introduction) can be accessed on http://localhost:3000/api/hello. This endpoint can be edited in `pages/api/hello.js`.

The `pages/api` directory is mapped to `/api/*`. Files in this directory are treated as [API routes](https://nextjs.org/docs/api-routes/introduction) instead of React pages.

Learn More

To learn more about Next.js, take a look at the following resources:

- [Next.js Documentation](https://nextjs.org/docs) - learn about Next.js features and API.
- [Learn Next.js](https://nextjs.org/learn) - an interactive Next.js tutorial.

You can check out [the Next.js GitHub repository](https://github.com/vercel/next.js/) - your feedback and contributions are welcome!

Deploy on Vercel

The easiest way to deploy your Next.js app is to use the [Vercel Platform](https://vercel.com/new?utm_medium=default-template&filter=next.js&utm_source=create-next-app&utm_campaign=create-next-app-readme) from the creators of Next.js.

Check out our [Next.js deployment documentation](https://nextjs.org/docs/deployment) for

more details.

File: Experience.js

```
import { forwardRef, useEffect, useRef, useState } from "react";

import ExperienceTabBar from "../ExperienceTabBar";
import Typography from "@mui/material/Typography";
import Grid from "@mui/material/Grid";
import StuySU from "../tabs/StuySU";
import BlockchainsForSchools from "../tabs/BlockchainsForSchools";
import Slide from "@mui/material/Slide";
import ArrowDownward from "@mui/icons-material/ArrowDownward";
import StuyBOE from "../tabs/StuyBOE";
import StellarCellarDoors from "../tabs/StellarCellarDoors";
import styles from "../../styles/Home.module.css";

function ExperienceWithRef({ backdropRef, experienceRef }, ref) {
  const [tab, setTab] = useState("stuysu");
  const containerRef = useRef();
  const [display, setDisplay] = useState(false);
  const [observing, setObserving] = useState(false);

  useEffect(() => {
    if (backdropRef && backdropRef.current && !observing) {
      const options = {
        threshold: 0.9,
      };

      const callback = (entries) => {
        setDisplay(!entries[0].isIntersecting);
      };

      const observer = new IntersectionObserver(callback, options);
      observer.observe(backdropRef.current);

      setObserving(true);
    }
  }, [backdropRef, observing, containerRef]);

  return (
    <div
      style={{
        minHeight: 600,
        transition: "height 0.5s ease-in",
      }}
      ref={ref}
    >
      <Typography
        align={"center"}
        variant={"body1"}
        sx={{
```



```

opacity: display ? 0 : 1,
transition: "opacity 0.5s ease-in",
cursor: "pointer",
}}
className={styles.bouncingText}
onClick={() =>
window.scrollTo({
left: 0,
top: window.innerHeight * 0.8,
behavior: "smooth",
})
}
>
<ArrowDownward
sx={{
verticalAlign: "middle",
margin: 1,
}}
/>
Keep Scrolling
</Typography>

<Slide
in={display}
direction={display ? "up" : "left"}
unmountOnExit
mountOnEnter
timeout={200}
container={experienceRef.current}
>
<div>
<Typography variant={"h3"} align={"center"}>
Experience
</Typography>
<Typography
variant={"subtitle1"}
color="text.secondary"
align={"center"}
gutterBottom
>
I&apos;m known to dabble here and there
</Typography>

<Grid container sx={{ margin: "1rem 0" }}>
<Grid item xs={12} sm={4} md={3} lg={3} xl={3}>
<ExperienceTabBar value={tab} setValue={setTab} />
</Grid>

<Grid
item
xs={12}
sm={8}
md={9}
lg={9}

```

```

xl={9}
ref={containerRef}
sx={{ overflowY: "clip" }}
>
<StuySU container={containerRef.current} tab={tab} />
<StuyBOE tab={tab} container={containerRef.current} />
<BlockchainsForSchools
  container={containerRef.current}
  tab={tab}
/>
<StellarCellarDoors tab={tab} container={containerRef.current} />
</Grid>
</Grid>
</div>
</Slide>
</div>
);
}

```

```
const Experience = forwardRef(ExperienceWithRef);
```

```
export default Experience;
```

File: ExperienceCard.js

```

import Card from "@mui/material/Card";
import CardHeader from "@mui/material/CardHeader";
import CardMedia from "@mui/material/CardMedia";
import CardContent from "@mui/material/CardContent";
import IconButton from "@mui/material/IconButton";
import Typography from "@mui/material/Typography";

import GitHub from "@mui/icons-material/GitHub";
import Chip from "@mui/material/Chip";
import Star from "@mui/icons-material/Star";
import Link from "@mui/material/Link";

export default function ExperienceCard({
  title,
  image,
  alt,
  content,
  tags = [],
  url,
  href,
  github,
}) {
  return (
    <Card>
      <CardHeader
        title={title}
        subheader={

```

```

!!href && (
<Link
href={href}
rel={"noopener noreferrer"}
variant={"subtitle2"}
target={"_blank"}
>
{url || href}
</Link>
)
}
action={
!!github && (
<IconButton
href={github}
target={"_blank"}
rel={"noopener noreferrer"}
>
<GitHub />
</IconButton>
)
}
/>
<CardMedia component="img" height="250" image={image} alt={alt} />
<CardContent>
<Typography variant="body2" color="text.secondary">
{content}
</Typography>

<div>
{tags.map((tag) => (
<Chip
label={tag}
key={tag}
variant={"outlined"}
icon=<Star sx={{ fontSize: 16 }} />
color={"secondary"}
sx={{ margin: "10px 10px 0 0", fontSize: 12 }}
/>
))}
</div>
</CardContent>
</Card>
);
}

```

File: ExperienceTabBar.js

```

import Tabs from "@mui/material/Tabs";
import Tab from "@mui/material/Tab";
import Box from "@mui/material/Box";

```

```

export default function ExperienceTabBar({ value, setValue }) {
  return (
    <Box
      sx={{
        bgcolor: "background.paper",
        margin: "10px 0",
      }}
    >
      <Tabs
        orientation="vertical"
        variant="scrollable"
        value={value}
        onChange={(_, t) => setValue(t)}
        aria-label="Organizations"
        sx={{ borderRight: 1, borderColor: "divider", textAlign: "right" }}
      >
        <Tab label="Stuyvesant Student Union" value={"stuysu"} />
        <Tab label="Stuyvesant Board of Elections" value={"stuyboe"} />
        <Tab label="BlockChains for Schools" value={"bfs"} />
        <Tab label="Stellar Cellar Doors" value={"stellar-doors"} />
      </Tabs>
    </Box>
  );
}

```

File: ExperienceTabPanel.js

```

import Typography from "@mui/material/Typography";
import Slide from "@mui/material/Slide";
import { useEffect, useState } from "react";
import ExperienceCard from "../ExperienceCard";
import Grid from "@mui/material/Grid";

export default function ExperienceTabPanel({
  container,
  isActive,
  name,
  projects,
}) {
  const [display, setDisplay] = useState(false);

  useEffect(() => {
    if (isActive) {
      const timeout = setTimeout(() => {
        setDisplay(true);
      }, 200);
    }

    return () => clearTimeout(timeout);
  });

  setDisplay(false);
}, [isActive]);

```

```

return (
  <Slide
    in={display}
    mountOnEnter
    unmountOnExit
    direction={display ? "up" : "left"}
    timeout={{ enter: 500, exit: 200, appear: 200 }}
    container={container}
  >
    <div>
      <Typography
        variant={"h4"}
        align={"center"}
        color={"primary"}
        gutterBottom
      >
        {name}
      </Typography>

      <Grid container padding={2} spacing={2}>
        {projects.map((project, index) => (
          <Grid item xs={12} sm={12} md={6} lg={6} xl={6} key={index}>
            <ExperienceCard {...project} />
          </Grid>
        ))}
      </Grid>
    </div>
  </Slide>
);
}

```

File: BlockchainsForSchools.js

```

import ExperienceTabPanel from "../ExperienceTabPanel";

const projects = [
  {
    title: "HackBFS",
    image: "/hackbfs.com_.png",
    alt: "HackBFS Home",
    content:
      "Designed, created, and set up infrastructure to serve the hackathon website for thousands of visitors",
    tags: ["ReactJS"],
    url: "hackbfs.com",
    href: "https://hackbfs.com",
    github: "https://github.com/blockchainsforschools/ideation-challenge",
  },
  {
    title: "Main Site",
    image: "/blockchainsforschools.org_.png",

```

```

alt: "Blockchains for Schools Home",
content:
"Led a team of developers and designers to decrease bounce rates and improve session
times on the main site.",
tags: ["ExpressJS", "PostgreSQL"],
url: "blockchainsforschools.org",
href: "https://blockchainsforschools.org",
github: "https://github.com/blockchainsforschools/main-site",
},
];

export default function BlockchainsForSchools({ container, tab }) {
const isActive = tab === "bfs";

return (
<ExperienceTabPanel
container={container}
isActive={isActive}
name={"Blockchains for Schools"}
projects={projects}
/>
);
}

```

File: StellarCellarDoors.js

```

import ExperienceTabPanel from "../ExperienceTabPanel";

const projects = [
{
title: "Stellar Doors Site",
image: "/stellarcellardoors.com_.png",
alt: "Stellar Cellar Doors home",
content:
"Designed and created a site for a cellar door business from scratch with a built in CMS
for easy updates.",
tags: ["Next.JS", "GraphQL", "MongoDB"],
url: "stellarcellardoors.com",
href: "https://stellarcellardoors.com",
},
];

export default function StellarCellarDoors({ tab, container }) {
const isActive = tab === "stellar-doors";

return (
<ExperienceTabPanel
projects={projects}
isActive={isActive}
container={container}
name={"Stellar Cellar Doors"}
/>

```

```
);  
}
```

File: StuyBOE.js

```
import ExperienceTabPanel from "../ExperienceTabPanel";  
  
const projects = [  
  {  
    title: "Voting Site",  
    image: "/vote.stuysu.org_.png",  
    alt: "Voting Site Home",  
    content:  
      "Enabled Student Government elections to happen remotely while ensuring security,  
      anonymity, and improving voter turnout by 107 percent in the first year. Also designed  
      the new logo.",  
    tags: ["Next.JS", "GraphQL", "MongoDB"],  
    url: "vote.stuysu.org",  
    href: "https://vote.stuysu.org",  
    github: "https://github.com/abir-taheer/vote.stuysu.org",  
  },  
];  
  
export default function StuyBOE({ tab, container }) {  
  const isActive = tab === "stuyboe";  
  
  return (  
    <ExperienceTabPanel  
      projects={projects}  
      isActive={isActive}  
      container={container}  
      name={"Stuyvesant Board of Elections"}  
    />  
  );  
}
```

File: StuySU.js

```
import ExperienceTabPanel from "../ExperienceTabPanel";  
  
const projects = [  
  {  
    title: "StuyActivities",  
    image: "/stuyactivities.org_.png",  
    alt: "StuyActivities Home",  
    content:  
      "Led the creation of one of the most powerful and scalable organization management  
      systems at one of the largest high schools in the United States",  
    tags: ["NodeJS", "ReactJS", "GraphQL", "Google One Tap"],  
    url: "stuyactivities.org",  
  },  
];
```

```

href: "https://stuyactivities.org",
github: "https://github.com/stuysu/stuyactivities.org",
},
{
title: "Student Union Applications",
image: "/applications.stuysu.org_.png",
alt: "Applications Site Home",
content:
"Developed the methodology and site for using modern cryptography and Google Drive APIs
to allow students to anonymously apply for Student Union positions.",
tags: [
"WebCrypto API",
"Next.JS",
"GraphQL",
"Google Drive API",
"MongoDB",
],
url: "applications.stuysu.org",
href: "https://applications.stuysu.org",
github: "https://github.com/stuysu/applications.stuysu.org",
},
{
title: "Valentines Day Letters",
image: "/valentines.stuysu.org_.png",
alt: "Valentines Letters Site Home",
content:
"A site to let students send each other valentines day cards virtually with the option
of anonymity, all implemented alongside a toxicity filter",
tags: ["Next.JS", "GraphQL", "MongoDB", "Google Drive API"],
href: "https://valentines.stuysu.org",
url: "valentines.stuysu.org",
github: "https://github.com/stuysu/valentines.stuysu.org",
},
];

export default function StuySU({ container, tab }) {
const isActive = tab === "stuysu";

return (
<ExperienceTabPanel
container={container}
isActive={isActive}
name={"Stuyvesant Student Union"}
projects={projects}
/>
);
}

```

File: Devpost.js

```
import SvgIcon from "@mui/material/SvgIcon";
```



```

export default function Devpost() {
  return (
    <SvgIcon
      x="0px"
      y="0px"
      xmlns="http://www.w3.org/2000/svg"
      viewBox="0 0 222 192.22"
    >
      <path
        className="cls-1"
        d="M135,85.67H122.69V157h11.48c24.47,0,35.72-14.34,35.72-35.72C170,97.55,159.75,85.67,135,85.67Z"
        transform="translate(-29.15 -25.24)"
      />
      <path
        className="cls-1"
        d="M195.63,25.24h-111L29.15,121.39l55.52,96.07h111l55.52-96.07ZM134.25,179.68H99.4V63h36.51C169.26,63,194,78.7,194,121.39,194,162.42,164.27,179.68,134.25,179.68Z"
        transform="translate(-29.15 -25.24)"
      />
    </SvgIcon>
  );
}

```

File: StackOverflow.js

```

import SvgIcon from "@mui/material/SvgIcon";

export default function StackOverflow() {
  return (
    <SvgIcon x="0px" y="0px" viewBox="0 0 8192 8192">
      <path
        className="st0"
        d="M7157.3,2982.8l-493.6,643.1l0,0l-342.3,734.4l-0.1,0l-175.5,791.1l-3956.4-877.1l0,0l175.5-791.1l3954.8,876.6
        L2648.5,2647.7v0l342.5-734.7l3670.9,1711.6L3448.3,1158.7l356.6-464.8h368.9L7157.3,2982.8
        L7157.3,2982.8z M2093.3,5151.8h4052.5
        v810.5l-4052.5-0.1v5151.8z"
      />
      <path
        className="st0"
        d="M7338.5,5440.2V7872h-6485V5440.2h810.6v1621.2h4863.8V5440.2H7338.5z"
      />
    </SvgIcon>
  );
}

```

File: Icons.js

```

import styles from "../Icons.module.css";

```

```

import IconButton from "@mui/material/IconButton";
import GitHub from "@mui/icons-material/GitHub";
import LinkedIn from "@mui/icons-material/LinkedIn";
import StackOverflow from "../icons/StackOverflow";
import Devpost from "../icons/Devpost";
import { Instagram } from "@mui/icons-material";

const icons = [
  {
    icon: <Devpost />,
    platform: "Devpost",
    url: "https://devpost.com/abir-taheer",
  },
  {
    icon: <StackOverflow />,
    platform: "StackOverflow",
    url: "https://stackoverflow.com/users/10237430/abir-taheer",
  },
  {
    icon: <GitHub />,
    platform: "GitHub",
    url: "https://github.com/abir-taheer",
  },
  {
    icon: <LinkedIn />,
    platform: "LinkedIn",
    url: "https://linkedin.com/in/AbirTaheer",
  },
  {
    icon: <Instagram />,
    platform: "Instagram",
    url: "https://instagram.com/abir.taheer",
  },
];

export default function Icons() {
  return (
    <div className={styles.container}>
      {icons.map(({ url, platform, icon }) => (
        <IconButton
          key={platform}
          sx={{
            color: "rgba(255, 255, 255, 0.6)",
            margin: "1rem",
            ":hover": {
              color: "white",
            },
          }}
          "@media (max-width: 500px)": {
            margin: "0.5rem",
          },
          href={url}
          target={"_blank"}
          rel={"noopener noreferrer"}

```

```
aria-label={platform + " Profile"}
>
{icon}
</IconButton>
)}}
</div>
);
}
```

File: Icons.module.css

```
.container {
display: flex;
align-content: center;
justify-content: center;
align-items: center;
}
```

File: LightroomSlideshow.js

```
import Typography from "@mui/material/Typography";
import { useEffect, useState } from "react";
import Slide from "@mui/material/Slide";

export default function LightroomSlideshow({ slideshowRef }) {
  const [display, setDisplay] = useState(false);
  const [observing, setObserving] = useState(false);

  useEffect(() => {
    if (slideshowRef && slideshowRef.current && !observing) {
      const options = {
        threshold: 0.1,
      };

      const callback = (entries) => {
        const entry = entries[0];
        setDisplay(entry.isIntersecting);
      };

      const observer = new IntersectionObserver(callback, options);
      observer.observe(slideshowRef.current);

      setObserving(true);
    }
  }, [slideshowRef, observing]);

  return (
    <Slide
      in={display}
      mountOnEnter
```

```

unmountOnExit
direction={display ? "up" : "left"}
timeout={{ enter: 500, exit: 200, appear: 200 }}
container={slideshowRef.current}
>
<div>
<Typography variant={"h3"} align={"center"} gutterBottom>
Photography
</Typography>
<Typography
variant={"subtitle1"}
align={"center"}
color={"text.secondary"}
>
I'm working on making a better gallery component but for the time
being here's a lightroom slideshow of my photos instead
</Typography>
<iframe
id="iframe"
src="https://lightroom.adobe.com/embed/shares/bf5594d5c8394001a7e972c1b03f2379/slideshow
?background_color=%232D2D2D&color=%23999999"
frameBorder={0}
style={{
width: "100%",
height: "570px",
}}
/>
</div>
</Slide>
);
}

```

File: ThemeProvider.js

```

import theme from "../theme";
import { ThemeProvider as Provider } from "@mui/material/styles";

export default function ThemeProvider({ children }) {
return <Provider theme={theme}>{children}</Provider>;
}

```

File: theme.js

```

import { createTheme } from "@mui/material/styles";

const theme = createTheme({
typography: {
fontFamily: `"Poppins", sans-serif`,
},
});

```

```
export default theme;
```

Repository: abir.taheer.me-old

File: README.md

This is a [Next.js](https://nextjs.org/) project bootstrapped with [`create-next-app`](https://github.com/vercel/next.js/tree/canary/packages/create-next-app).

Getting Started

First, run the development server:

```
``bash
npm run dev
# or
yarn dev
``
```

Open http://localhost:3000 with your browser to see the result.

You can start editing the page by modifying `pages/index.js`. The page auto-updates as you edit the file.

[API routes](https://nextjs.org/docs/api-routes/introduction) can be accessed on http://localhost:3000/api/hello. This endpoint can be edited in `pages/api/hello.js`.

The `pages/api` directory is mapped to `/api/*`. Files in this directory are treated as [API routes](https://nextjs.org/docs/api-routes/introduction) instead of React pages.

Learn More

To learn more about Next.js, take a look at the following resources:

- [Next.js Documentation](https://nextjs.org/docs) - learn about Next.js features and API.
- [Learn Next.js](https://nextjs.org/learn) - an interactive Next.js tutorial.

You can check out [the Next.js GitHub repository](https://github.com/vercel/next.js/) - your feedback and contributions are welcome!

Deploy on Vercel

The easiest way to deploy your Next.js app is to use the [Vercel Platform](https://vercel.com/import?utm_medium=default-template&filter=next.js&utm_source=create-next-app&utm_campaign=create-next-app-readme) from the creators of Next.js.

Check out our [Next.js deployment documentation](https://nextjs.org/docs/deployment) for more details.

File: PictureGrid.js

```
import $ from "jquery";
import "justifiedGallery";
import { createRef, useEffect } from "react";
import "justifiedGallery/dist/css/justifiedGallery.min.css";

const PictureGrid = ({ pictures }) => {
  const galleryRef = createRef();

  useEffect(() => {
    if (galleryRef.current) {
      $(galleryRef.current).justifiedGallery({
        maxHeight: 500,
        rowHeight: 250,
        margins: 5,
        randomize: false,
        lastRow: "justify",
        captions: false,
        justifyThreshold: 0.6,
      });
    }
  });

  return (
    <div ref={galleryRef}>
      {pictures.map(picture => (
        <a key={picture.id}>
          <img
            alt={picture.resource.tags.join(", ")}
            src={picture.resource.url}
            height={picture.resource.height}
            width={picture.resource.width}
          />
        </a>
      ))}
    </div>
  );
};

export default PictureGrid;
```

File: graphqlClient.js

```
import { ApolloClient, InMemoryCache } from "@apollo/client";

export const cache = new InMemoryCache();

export const graphqlClient = new ApolloClient({
  uri: "/api/graphql",
  cache,
});
```

```
export default graphqlClient;
```

File: cover.js

```
import CloudinaryResource from "../../../models/cloudinaryResource";

export default album => CloudinaryResource.idLoader.load(album.coverPicId);
```

File: index.js

```
import albums from "./albums";
import albumById from "./albumById";
import albumByUrl from "./albumByUrl";
import people from "./people";
import personById from "./personById";
import pictureById from "./pictureById";
import picturesByAlbumId from "./picturesByAlbumId";
import picturesByAlbumUrl from "./picturesByAlbumUrl";
```

```
const Query = {
  albumById,
  albumByUrl,
  albums,
  people,
  personById,
  pictureById,
  picturesByAlbumId,
  picturesByAlbumUrl,
};
```

```
export default Query;
```

File: pictures.js

```
import Picture from "../../../models/picture";

export default album =>
  Promise.all(album.pictureIds.map(id => Picture.idLoader.load(id)));
```

File: url.js

```
import { v2 as cloudinary } from "cloudinary";

export default (resource, { preset }) => {
  if (preset === "previewLarge") {
```



```
return cloudinary.url(resource._id, {
  secure: true,
  sign_url: true,
  height: 800,
  quality: 80,
});
}
```

```
if (preset === "thumbnailMedium") {
  return cloudinary.url(resource._id, {
    secure: true,
    sign_url: true,
    height: 540,
  });
}
```

```
if (preset === "thumbnailSmall") {
  return cloudinary.url(resource._id, {
    secure: true,
    sign_url: true,
    height: 260,
    quality: 90,
  });
}
};
```

File: name.js

```
export default person => person.firstName + " " + person.lastName;
```

File: picture.js

```
import CloudinaryResource from "../../models/cloudinaryResource";

export default person => CloudinaryResource.idLoader.load(person.pictureId);
```

File: albums.js

```
import Album from "../../models/album";

export default () => Album.find();
```

File: people.js

```
import Person from "../../models/person";
```

```
export default () => Person.find();
```

File: resource.js

```
import CloudinaryResource from "../../models/cloudinaryResource";

export default picture => CloudinaryResource.idLoader.load(picture.resourceId);
```

File: albumById.js

```
import Album from "../../models/album";

export default (_, { id }) => Album.findById(id);
```

File: albumByUrl.js

```
import Album from "../../models/album";

export default (_, { url }) => Album.findOne({ url });
```

File: Album.js

```
import { gql } from "apollo-server-micro";

export default gql`
type Album {
  id: ObjectId!
  url: String
  title: String
  description: String

  # Dynamic Props
  cover: CloudinaryResource
  pictures: [Picture]
}
`;
```

File: CloudinaryResource.js

```
import { gql } from "apollo-server-micro";

export default gql`
enum UrlPresets {
  thumbnailSmall
```

```

    thumbnailMedium
    previewLarge
  }

  type CloudinaryResource {
    id: String
    width: Int
    height: Int
    format: String
    resourceType: String
    createdAt: DateTime
    tags: [String]

    # Dynamic props
    url(preset: UrlPresets!): String
  }
`;

```

File: Person.js

```

import { gql } from "apollo-server-micro";

export default gql`
  type SocialMedia {
    facebook: String
    twitter: String
    email: String
    instagram: String
    linkedIn: String
    github: String
  }

  type Person {
    id: ObjectId
    firstName: String
    lastName: String
    social: SocialMedia

    # Dynamic props
    name: String
    picture: CloudinaryResource
  }
`;

```

File: Picture.js

```

import { gql } from "apollo-server-micro";

export default gql`
  type Picture {

```

```
id: ObjectId!
title: String
description: String
takenAt: DateTime

# Dynamic Props ----
# Cloudinary Resource Object to facilitate access
resource: CloudinaryResource
# The people in the photo
people: [Person]

# Albums that contain this photo
albums: [Album]
}
`;
```

File: Query.js

```
import { gql } from "apollo-server-micro";

export default gql`
type Query {
  albums: [Album]
  albumById(id: ObjectId!): Album
  albumByUrl(url: String!): Album

  picturesByAlbumId(albumId: ObjectId!): [Picture]
  picturesByAlbumUrl(albumUrl: String!): [Picture]
  pictureById(id: ObjectId!): Picture

  people: [Person]
  personById(id: ObjectId!): Person
}
`;
```

Repository: adhan-pi

File: app.py

```
import json
import os
import re
import time
import sys
import psutil
import logging
from datetime import datetime

def update_times():
    import requests
    api_data = requests.get(
        "http://api.aladhan.com/v1/timingsByCity?city=New+York&country=United+States&method=2").
    json()
    ptime = api_data["data"]["timings"]
    stored_data = {
        "date": today,
        "times": {
            "Fajr": {
                "time": ptime["Fajr"],
                "done": False
            },
            "Dhuhr": {
                "time": ptime["Dhuhr"],
                "done": False
            },
            "Asr": {
                "time": ptime["Asr"],
                "done": False
            },
            "Maghrib": {
                "time": ptime["Maghrib"],
                "done": False
            },
            "Isha": {
                "time": ptime["Isha"],
                "done": False
            },
        }
    }
    f = open("times.json", "w")
    f.write(json.dumps(stored_data))
    f.close()
    return stored_data

current_dir = os.path.dirname(os.path.realpath(__file__))
```

```

while not time.sleep(60):

today = datetime.today().strftime('%Y-%m-%d')
try:
time_json = open("times.json", "r").read()
times = json.loads(time_json)
assert (times["date"] == today), "Prayer times are not up to date"
except:
times = update_times()

adhan_made = True
max_time = None

for prayer in times["times"]:
prayer_time = datetime.strptime(times["times"][prayer]["time"], '%H:%M').time()
if prayer_time < datetime.today().time() and (max_time is None or prayer_time >
max_time):
current_prayer = prayer
adhan_made = times["times"][prayer]["done"]
max_time = prayer_time

if not adhan_made:
is_fajr = current_prayer == "Fajr"
times["times"][current_prayer]["done"] = True
f = open("times.json", "w")
f.write(json.dumps(times))
f.close()

cec_resp = os.popen('echo pow 0 | cec-client -s -d 1').read()
status_search = re.finditer(r"(?<=\b(power\sstatus:)\s)(\w+)", cec_resp)
for x in status_search:
start = int(x.start())
end = int(x.end())

status = cec_resp[start:end]
is_on = status == "on"

if not is_on:
os.system("echo on 0 | cec-client -s -d 1")
time.sleep(3)

os.system('echo "as" | cec-client RPI -s -d 1')
time.sleep(2)

adhan_name = "fajr_adhan.mp3" if is_fajr else "standard_adhan.mp3"
adhan_length = 185
os.system("omxplayer --no-keys " + adhan_name + " &")

time.sleep(adhan_length)

os.system("omxplayer --no-keys after_adhan.mp3 &")

os.system("clear")

```

```
if not is_on:  
    time.sleep(20)  
    os.system("echo standby 0 | cec-client -s -d 1")
```

Repository: advent-of-code-2021

File: README.md

advent-of-code-2021

File: day1.js

```
const getInput = require("./getInput");

async function parseInput() {
  const input = await getInput(1);
  return input.map(Number);
}

async function part1() {
  const input = await parseInput();

  let count = 0;

  input.forEach((num, i) => {
    if (i && num > input[i - 1]) {
      count++;
    }
  });

  return count;
}

async function part2() {
  const input = await parseInput();

  let count = 0;

  input.forEach((num, i) => {
    let first = input.slice(i - 3, i);
    let second = input.slice(i - 2, i + 1);

    if (first.length && second.length) {
      const firstSum = first.reduce((a, b) => a + b);
      const secondSum = second.reduce((a, b) => a + b);

      if (secondSum > firstSum) {
        count++;
      }
    }
  });

  return count;
}
```


File: day2.js

```
const getInput = require("../getInput");
```

```
async function parseInput() {  
  const input = await getInput(2);
```

```
  return input.map((i) => {  
    const a = i.split(" ");  
  
    a[1] = Number(a[1]);  
  });  
}
```

```
async function part1() {  
  const input = await parseInput();
```

```
  let depth = 0;  
  let horizontal = 0;
```

```
  input.forEach((a) => {  
    if (a[0] === "forward") {  
      horizontal += a[1];  
    }
```

```
    if (a[0] === "down") {  
      depth += a[1];  
    }
```

```
    if (a[0] === "up") {  
      depth -= a[1];  
    }  
  });
```

```
  return depth * horizontal;  
}
```

```
async function part2() {  
  const input = await getInput(2);
```

```
  let h = 0;  
  let d = 0;  
  let a = 0;
```

```
  input.forEach((i) => {  
    if (i[0] === "forward") {  
      h += i[1];  
      d += a * i[1];  
    }
```

```
    if (i[0] === "down") {  
      a += i[1];  
    }
```

```
}

if (i[1] === "up") {
  a -= i[1];
}
});

return h * d;
}
```

Repository: advent-of-code-2022

File: README.md

Advent of Code 2022

Welcome to my repository for the 2022 Advent of Code challenge! Advent of Code is an annual event where programmers around the world attempt to solve a series of small programming challenges, one for each day in the month of December leading up to Christmas.

This repository contains my solutions to the Advent of Code challenges, written in TypeScript. I chose to use TypeScript for these solutions because I wanted to prioritize code readability and structure in my solutions rather than leaderboard performance.

Each solution is contained in its own file in the `src/days/` directory, named according to the day of the Advent of Code challenge it corresponds to. Within each file, you will find the TypeScript source code for my solutions to both parts. There's also a `utils` directory with accompanying tools and helper functions that I wrote to make it easier for me to solve the challenges.

There's also a helper function in the `utils` directory called [`getInput`](https://github.com/abir-taheer/advent-of-code-2022/blob/main/src/utils/getInput.ts) that automatically fetches your problem input for the day that you provide as an argument from the Advent of Code website.

1. It uses the value of your session cookie to authenticate as you and fetch the correct input.
2. You can find your session cookie by [inspecting the network requests](https://superuser.com/questions/1486002/how-do-i-see-request-cookies-in-chrome) made by your browser when you visit the Advent of Code website.
3. Once you find it, set it as the value of the `SESSION_COOKIE` environment variable. You can add a `.env` file to the root of the project and set the value there.

I hope that my solutions will be helpful to you as you work on the Advent of Code challenges, or as you simply try to improve your skills in programming. If you have any questions or comments, feel free to reach out to me through the repository's issue tracker.

Happy coding!

File: constants.ts

```
import { config } from "dotenv";

config();

export const AOC_SESSION_ID = process.env.AOC_SESSION_ID || "";
```

File: day1.ts

```
import { getInput } from "../utils/getInput";
import { sum } from "../utils/sum";

const parseInput = async () => {
  const raw = await getInput(1);
  return raw.split("\n\n").map((a) => a.split("\n").map(Number));
};

const part1 = async () => {
  const input = await parseInput();

  return Math.max(...input.map((elf) => sum(elf)));
};

const part2 = async () => {
  const input = await parseInput();

  return sum(
    input
      .map((elf) => sum(elf))
      .sort((a, b) => b - a)
      .slice(0, 3)
  );
};

part1().then((p1) => console.log({ p1 }));
part2().then((p2) => console.log({ p2 }));
```

File: day2.ts

```
import { getInput } from "../utils/getInput";

const parseInput = async () => {
  const raw = await getInput(2);

  return raw
    .split("\n")
    .filter(Boolean)
    .map((a) => a.split(" "));
};

const part1 = async () => {
  const input = await parseInput();

  let score = 0;

  input.forEach(([opponent, you]) => {
    if (you === "X") {
```

```

score += 1;

if (opponent === "A") {
score += 3;
} else if (opponent === "C") {
score += 6;
}
}

if (you === "Y") {
score += 2;

if (opponent === "B") {
score += 3;
} else if (opponent === "A") {
score += 6;
}
}

if (you === "Z") {
score += 3;

if (opponent === "C") {
score += 3;
} else if (opponent === "B") {
score += 6;
}
}
});

return score;
};

const part2 = async () => {
const input = await parseInput();

let score = 0;

input.forEach(([opponent, you]) => {
if (you === "X") {
// you lost
score += opponent === "A" ? 3 : opponent === "B" ? 1 : 2;
} else if (you === "Y") {
score += 3;

score += opponent === "A" ? 1 : opponent === "B" ? 2 : 3;
} else if (you === "Z") {
score += 6;

score += opponent === "A" ? 2 : opponent === "B" ? 3 : 1;
}
});

return score;

```

```
};
```

```
part1().then((p1) => console.log({ p1 }));  
part2().then((p2) => console.log({ p2 }));
```

File: day3.ts

```
import { getInput } from "../utils/getInput";  
import { groupItems } from "../utils/groupItems";  
  
const day = 3;  
  
const parseInput = async (numSacks: number) => {  
  const raw = await getInput(day);  
  
  return raw  
    .split("\n")  
    .filter(Boolean)  
    .map((row) =>  
      Array.from(Array(numSacks), (_, i) =>  
        row.slice(  
          Math.ceil((row.length * i) / numSacks),  
          Math.ceil((row.length * (i + 1)) / numSacks)  
        )  
      )  
    );  
};  
  
const getLetterScore = (letter: string) => {  
  let bonus = 0;  
  if (letter.toUpperCase() === letter) {  
    bonus += 26;  
  }  
  
  return letter.toLowerCase().charCodeAt(0) - 96 + bonus;  
};  
  
const part1 = async () => {  
  const input = await parseInput(2);  
  
  let sum = 0;  
  input.forEach(([left, right]) => {  
    const leftCharSet = new Set(left);  
    const rightCharSet = new Set(right);  
    const common = Array.from(leftCharSet).filter((char) =>  
      rightCharSet.has(char)  
    );  
  
    sum += getLetterScore(common[0]);  
  });  
  
  return sum;
```

```

};

const part2 = async () => {
const input = await parseInput(1);

let sum = 0;
groupItems(input, 3).forEach([first, second, third]) => {
const firstCharSet = new Set(first);
const secondCharSet = new Set(second);
const thirdCharSet = new Set(third);

const common = Array.from(firstCharSet).filter(
(char) => secondCharSet.has(char) && thirdCharSet.has(char)
);

sum += getLetterScore(common[0]);
});

return sum;
};

part1().then((p1) =>
console.log({
p1,
}))
);
part2().then((p2) => console.log({ p2 }));

```

File: day4.ts

```

import { getInput } from "../utils/getInput";
import { oneRangeContainsAnther, rangeHasOverlap } from "../utils/range";

const day = 4;

const parseInput = async () => {
const raw = await getInput(day);
return raw
.split("\n")
.filter(Boolean)
.map((a) => a.split(",").map((b) => b.split("-").map(Number) as Range));
};

type Range = [number, number];

const part1 = async () => {
const input = await parseInput();
const onesWithOverlap = input.filter((row) =>
oneRangeContainsAnther(row[0], row[1])
);
return onesWithOverlap.length;
};

```

```

const part2 = async () => {
const input = await parseInput();

const onesWithOverlap = input.filter((row) =>
rangeHasOverlap(row[0], row[1])
);

return onesWithOverlap.length;
};

part1().then((p1) => console.log({ p1 }));
part2().then((p2) => console.log({ p2 }));

```

File: day5.ts

```

import { getInput } from "../utils/getInput";

const day = 5;

const parseInput = async () => {
const raw = await getInput(day);

const [rawChart, rawInstructions] = raw.split("\n\n");
const horizontalChart = rawChart
.split("\n")
.map((line) => line.replace(/[0-9]/g, "").trim())
.filter(Boolean)
.map((line) => line.trim().split(/ {1,4}/g));

const chart = Array.from(Array(10), (_, i) =>
horizontalChart.map((a) => a[i - 1]).filter(Boolean)
);

const instructions = rawInstructions
.split("\n")
.filter(Boolean)
.map((a) =>
a
.split(" ")
.map(Number)
.filter((a) => !Number.isNaN(a))
);

return { chart, instructions };
};

const part1 = async () => {
const { chart, instructions } = await parseInput();

const chartCopy = JSON.parse(JSON.stringify(chart)) as typeof chart;

```



```

instructions.forEach(([amount, from, to]) => {
const items = chartCopy[from].splice(0, amount).reverse();

chartCopy[to].unshift(...items);
});

return chartCopy.map((a) => a[0]).filter(Boolean);
};

const part2 = async () => {
const { chart, instructions } = await parseInput();

const chartCopy = JSON.parse(JSON.stringify(chart)) as typeof chart;

instructions.forEach(([amount, from, to]) => {
const items = chartCopy[from].splice(0, amount);

chartCopy[to].unshift(...items);
});

return chartCopy.map((a) => a[0]).filter(Boolean);
};

part1().then((p1) => console.log({ p1 }));
part2().then((p2) => console.log({ p2 }));

```

File: Grid.ts

```

export type Direction = "up" | "down" | "left" | "right";
export type RotationDegree = 0 | 90 | 180 | 270;

export class Grid<T extends Cell<any>> {
private rows: T[][] = [];

getRows(): T[][] {
return this.rows;
}

get(row: number, col: number) {
const rowArr = this.rows[row];

if (!rowArr) {
return null;
}

return this.rows[row][col] || null;
}

getColumns(): T[][] {
return this.rows.map((row, rowIndex) =>
row.map((_, colIndex) => this.get(colIndex, rowIndex)!)
);
}

```

```

}

addCell(row: number, cell: T) {
  if (!this.rows[row]) {
    this.rows[row] = [];
  }

  this.rows[row].push(cell);
}

updateCellLocs() {
  this.rows.forEach((row, rowIndex) => {
    row.forEach((cell, colIndex) => cell.setLocation(rowIndex, colIndex));
  });
}

rotate(degrees: RotationDegree) {
  if (degrees === 90) {
    this.rows = this.getColumns().map((col) => col.reverse());
  }

  if (degrees === 180) {
    this.rows = this.rows.reverse().map((row) => row.reverse());
  }

  if (degrees === 270) {
    this.rows = this.getColumns().reverse();
  }

  this.updateCellLocs();

  return this.rows;
}

export type CellProps<T> = {
  value: T;
  row: number;
  col: number;
  grid: Grid<Cell<T>>;
};

export class Cell<T> {
  private value: T | null = null;
  private row: number = 0;
  private col: number = 0;
  private grid: Grid<Cell<T>> | null = null;

  constructor({ value, row, col, grid }: CellProps<T>) {
    this.value = value;
    this.row = row;
    this.col = col;
    this.grid = grid;
  }

```

```

setLocation(row: number, col: number) {
  this.row = row;
  this.col = col;
}

getValue() {
  return this.value!;
}

setValue(val: T) {
  this.value = val;
}

getAdjacent(dir: Direction) {
  const row =
    dir === "up" ? this.row - 1 : dir === "down" ? this.row + 1 : this.row;
  const col =
    dir === "right" ? this.col + 1 : dir === "left" ? this.col - 1 : this.col;

  return this.grid!.get(row, col);
}
}

```

File: filesystem.ts

```

import { sum } from "../sum";

export type Item = File | Directory;

export class Directory {
  private size: number = 0;
  private sizeInitialized: boolean = false;

  readonly items: { [name: string]: Item } = {};

  name: string = "";
  parent: Directory | null = null;

  constructor({ name, parent }: { name: string; parent: Directory | null }) {
    this.name = name;
    this.parent = parent;
  }

  addItem(file: Item) {
    this.items[file.name] = file;
  }

  getSize(): number {
    if (!this.sizeInitialized) {
      this.size = this.calculateSize();
      this.sizeInitialized = true;
    }
  }

```

```

}

return this.size;
}

calculateSize(): number {
return (
sum(this.GetFiles().map((file) => file.size)) +
sum(this.getDirectories().map((dir) => dir.getSize()))
);
}

getDirectories(): Directory[] {
return Object.keys(this.items)
.map((name) => this.items[name])
.filter((item) => item instanceof Directory) as Directory[];
}

getFiles(): File[] {
return Object.keys(this.items)
.map((name) => this.items[name])
.filter((item) => item instanceof File) as File[];
}
}

export class File {
readonly size: number = 0;
readonly name: string = "";
readonly parent: Directory | null = null;

constructor({
name,
size,
parent,
}: {
name: string;
size: number;
parent: Directory;
}) {
this.size = size;
this.name = name;
this.parent = parent;
}
}

export class Terminal {
private path: string = "/";
private root: Directory = new Directory({ name: "/", parent: null });
private head: Directory = this.root;

constructor() {}

cd(dir: string) {
if (dir === "/") {

```

```

this.path = "/";
this.head = this.root;
} else if (dir === "..") {
if (!this.head.parent) {
throw new Error("Cannot go up from root");
}

this.path = this.path.split("/").slice(0, -1).join("/");
this.head = this.head.parent;
} else {
const newHead = this.head.items[dir];
if (!newHead) {
throw new Error(`No such directory: ${dir}`);
}

if (newHead instanceof File) {
throw new Error(`Not a directory: ${dir}`);
}

this.path = `${this.path}/${dir}`;
this.head = newHead;
}
}

getHead(): Directory {
return this.head;
}

getRoot(): Directory {
return this.root;
}

getPath(): string {
return this.path;
}
}

```

File: getInput.ts

```

import axios from "axios";
import { AOC_SESSION_ID } from "../constants";

const cache: {
[day: number]: string;
} = {};

export const getInput = async (day: number) => {
if (cache[day]) {
return cache[day];
}

const { data } = await axios<string>(

```

```
`https://adventofcode.com/2022/day/${day}/input`,
{
headers: {
Cookie: `session=${AOC_SESSION_ID}`,
},
}
);

cache[day] = data;

return data;
};
```

File: groupItems.ts

```
export const groupItems = <T>(items: T[], numPerGroup: number) => {
const result: T[][] = [];

items.forEach((item, index) => {
const bucket = Math.floor(index / numPerGroup);
if (!result[bucket]) {
result[bucket] = [];
}

result[bucket].push(item);
});

return result;
};
```

File: range.ts

```
export type Range = [number, number];

export const rangeHasOverlap = (a: Range, b: Range) => {
const aContainsBStart = a[0] <= b[0] && a[1] >= b[0];
const aContainsBEnd = a[0] <= b[1] && a[1] >= b[1];
const bContainsAStart = b[0] <= a[0] && b[1] >= a[0];
const bContainsAEnd = b[0] <= a[1] && b[1] >= a[1];

return aContainsBStart || aContainsBEnd || bContainsAStart || bContainsAEnd;
};

export const oneRangeContainsAnother = (a: Range, b: Range) => {
const aContainsB = a[0] <= b[0] && a[1] >= b[1];
const bContainsA = b[0] <= a[0] && b[1] >= a[1];

return aContainsB || bContainsA;
};
```

Repository: algorithmic-playground

File: README.md

```
# algorithmic-playground
```

File: base64.py

```
# My own module that attempts to encode strings in base64
import copy

def int_to_binary(number: int, min_len=8) -> str:
    completed_string = ""
    # Make a copy so as not to affect the original element
    num_copy = copy.copy(number)

    # The factor is the largest power of two that fits into our number
    # It is essentially the value of the largest binary place
    factor = 1
    while num_copy >= factor * 2:
        factor *= 2

    # We're building up our string from the left side to the right
    while factor > 0:
        # This will always be 1 or 0
        # 1 means the factor is less than or equal to the current value of num_copy
        # 0 means the factor is greater than the value of num_copy
        completed_string += str(num_copy // factor)

    # This accounts for the current factor and reduces num_copy appropriately for the next
    # binary place check
    num_copy = num_copy % factor

    # Divide the factor by two to get the next value of the next binary place
    factor = factor // 2

    # Return the completed string after all of the 0's and 1's have been added
    # Pad the string with extra beginning 0's if it's shorter than the needed length
    return completed_string.rjust(min_len, '0')

# This string contains all characters used to encode in base64, in increasing order
base_64_chars = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"

def to_base_64(content: str) -> str:
    current_index = -1
    current_counter = 0
    binary_list = []
```

```

base_64_string = ""
for letter in str(content):
    if current_counter == 0:
        current_index += 1
        binary_list.append([])
        current_counter = (current_counter + 1) % 3
        binary_list[current_index].append(int_to_binary(ord(letter)))

    for x in range(3 - len(binary_list[current_index])):
        binary_list[current_index].append("00000000")

    for triplet_index in range(len(binary_list)):
        complete_string = binary_list[triplet_index][0] + binary_list[triplet_index][1] +
        binary_list[triplet_index][2]
        twenty_four_bits = [
            complete_string[:6],
            complete_string[6:12],
            complete_string[12:18],
            complete_string[18:],
        ]

        for bit_index in range(len(twenty_four_bits)):
            base_64_char_bin = twenty_four_bits[bit_index]
            char_index = bin_to_int(base_64_char_bin)
            if triplet_index == len(binary_list) - 1 and bit_index >= 2 and char_index == 0:
                base_64_string += "="
            else:
                base_64_string += base_64_chars[char_index]
        return base_64_string

def bin_to_int(b0: str) -> int:
    factor = 1
    num = 0
    for x in b0[::-1]:
        num += int(x) * factor
        factor *= 2
    return num

print(to_base_64('ABBBc cdsfsd'))

# TODO MAKE A TRANSLATOR THAT CONVERTS BASE64 BACK TO A STRING

```


Repository: async-rate-limit

File: README.md

```
# async-rate-limit

Queue setup for limiting things like api calls and other asynchronous tasks like api
calls that may take unpredictable amounts of time. Works with TypeScript.

## Usage:

### Importing
ES6 imports
```javascript
import RateLimit from "async-rate-limit";
```

CommonJS:
```javascript
const RateLimit = require("async-rate-limit");
```

### Using In practice
```javascript
// Pretending to have an API with a limit of 10 requests per second
const limiter = new RateLimit({limit: 10, timespan: 1000});

const sleep = (ms) => new Promise(resolve => setTimeout(resolve, ms));

for(let x = 0; x < 100; x++){
 limiter.perform(async () => {
 const requestTime = Math.floor(Math.random() * 2000);
 await sleep(requestTime);
 return requestTime;
 }).then(time => {
 console.log(`Completed Request ${x} in ${time}ms`);
 });
}
```

## Documentation

### Creating an instance of RateLimit:
```javascript
const limiter = new RateLimit(options);
```

### Options
Name:	Description:
limit	The maximum number of actions that can be performed in a given timespan.
```

Also the max number of concurrent events. Default: 5 |
| timespan | How long to wait before freeing up an action slot. Given in milliseconds.
Default: 1000. |

Methods

perform(action)

Takes in a function and returns a Promise.

The promise resolves to the value that is returned by calling the provided function. If the provided function returns a promise then it will wait for the promise to resolve first.

```
```javascript
```

```
const return5 = async () => 5;
```

```
limiter
```

```
.perform(return5)
```

```
.then(result => {
```

```
 console.log("Expected: 5; Result: " + result);
```

```
});
```

```
```
```

File: index.ts

```
const { queue } = require("async");
```

```
interface Props {
```

```
  limit?: number;
```

```
  timespan?: number;
```

```
}
```

```
class RateLimit {
```

```
  private readonly limit: number;
```

```
  private readonly timespan: number;
```

```
  private queue: any;
```

```
  static default: any;
```

```
  constructor({ limit, timespan }: Props) {
```

```
    this.limit = limit || 5;
```

```
    this.timespan = timespan || 1000;
```

```
    this.queue = queue((action: Function, cb: Function) => {
```

```
      action().then(() => setTimeout(cb, this.timespan));
```

```
    }, this.limit);
```

```
    this.perform = this.perform.bind(this);
```

```
  }
```

```
  perform(action: Function): Promise<any> {
```

```
    return new Promise((resolve, reject) => {
```

```
      this.queue.push(async () => {
```

```
try {  
  const result = await action();  
  resolve(result);  
} catch (er) {  
  reject(er);  
}  
});  
});  
}  
}
```

```
RateLimit.default = RateLimit;  
export = RateLimit;
```

Repository: athan

File: README.md

athan

plays the athan on local cast-enabled devices

[[Coverage
Status](<https://coveralls.io/repos/github/abir-taheer/athan/badge.svg?branch=main>)](<https://coveralls.io/github/abir-taheer/athan?branch=main>)

Repository: athan-pi

File: getDevices.js

```
"use strict";
var __importDefault = (this && this.__importDefault) || function (mod) {
  return (mod && mod.__esModule) ? mod : { "default": mod };
};
Object.defineProperty(exports, "__esModule", { value: true });
var fs_1 = __importDefault(require("fs"));
function getDevices() {
  var devices = JSON.parse(fs_1.default.readFileSync("./devices.json", "utf8"));
  return devices.map(function (row) {
    return {
      id: row.id,
      name: row.name,
      friendlyName: row.friendlyName,
      host: row.host,
      enabled: row.enabled,
      volume: row.volume,
      prayers: row.prayers,
    };
  });
}
exports.default = getDevices;
//# sourceMappingURL=getDevices.js.map
```

File: getPrayerTimesFromServer.js

```
"use strict";
var __awaiter = (this && this.__awaiter) || function (thisArg, _arguments, P, generator) {
  function adopt(value) { return value instanceof P ? value : new P(function (resolve) {
    resolve(value);
  }); }
  return new (P || (P = Promise))(function (resolve, reject) {
    function fulfilled(value) { try { step(generator.next(value)); } catch (e) { reject(e); } }
    function rejected(value) { try { step(generator["throw"](value)); } catch (e) { reject(e); } }
    function step(result) { result.done ? resolve(result.value) : adopt(result.value).then(fulfilled, rejected); }
    step((generator = generator.apply(thisArg, _arguments || [])).next());
  });
};
var __generator = (this && this.__generator) || function (thisArg, body) {
  var _ = { label: 0, sent: function() { if (t[0] & 1) throw t[1]; return t[1]; }, trys: [], ops: [] }, f, y, t, g;
  return g = { next: verb(0), "throw": verb(1), "return": verb(2) }, typeof Symbol === "function" && (g[Symbol.iterator] = function() { return this; }), g;
  function verb(n) { return function (v) { return step([n, v]); }; }
  function step(op) {
```

```

if (f) throw new TypeError("Generator is already executing.");
while (_) try {
if (f = 1, y && (t = op[0] & 2 ? y["return"] : op[0] ? y["throw"] || ((t = y["return"])
&& t.call(y), 0) : y.next) && !(t = t.call(y, op[1])).done) return t;
if (y = 0, t) op = [op[0] & 2, t.value];
switch (op[0]) {
case 0: case 1: t = op; break;
case 4: _.label++; return { value: op[1], done: false };
case 5: _.label++; y = op[1]; op = [0]; continue;
case 7: op = _.ops.pop(); _.trys.pop(); continue;
default:
if (!(t = _.trys, t = t.length > 0 && t[t.length - 1]) && (op[0] === 6 || op[0] === 2))
{ _ = 0; continue; }
if (op[0] === 3 && (!t || (op[1] > t[0] && op[1] < t[3]))) { _.label = op[1]; break; }
if (op[0] === 6 && _.label < t[1]) { _.label = t[1]; t = op; break; }
if (t && _.label < t[2]) { _.label = t[2]; _.ops.push(op); break; }
if (t[2]) _.ops.pop();
_.trys.pop(); continue;
}
op = body.call(thisArg, _);
} catch (e) { op = [6, e]; y = 0; } finally { f = t = 0; }
if (op[0] & 5) throw op[1]; return { value: op[0] ? op[1] : void 0, done: true };
};

var __importDefault = (this && this.__importDefault) || function (mod) {
return (mod && mod.__esModule) ? mod : { "default": mod };
};

Object.defineProperty(exports, "__esModule", { value: true });
var axios_1 = __importDefault(require("axios"));
var fs_1 = __importDefault(require("fs"));
function getPrayerTimesFromServer(date, city) {
return __awaiter(this, void 0, void 0, function () {
var timestamp, response, times;
return __generator(this, function (_a) {
switch (_a.label) {
case 0:
timestamp = Math.round(date.getTime() / 1000);
return [4 /*yield*/, axios_1.default.get("http://api.aladhan.com/v1/timingsByCity/" +
timestamp, {
params: {
city: city,
country: "US",
iso8601: true,
},
})];
case 1:
response = _a.sent();
times = {
date: date.toDateString(),
fajr: new Date(response.data.data.timings.Fajr),
dhuhr: new Date(response.data.data.timings.Dhuhr),
asr: new Date(response.data.data.timings.Asr),
maghrib: new Date(response.data.data.timings.Maghrib),
isha: new Date(response.data.data.timings.Isha),

```

```

};
fs_1.default.writeFileSync("./prayerTimes.json", JSON.stringify([times]));
return [2 /*return*/, times];
}
});
});
}
exports.default = getPrayerTimesFromServer;
//# sourceMappingURL=getPrayerTimesFromServer.js.map

```

File: initTables.js

```

"use strict";
var __importDefault = (this && this.__importDefault) || function (mod) {
return (mod && mod.__esModule) ? mod : { "default": mod };
};
Object.defineProperty(exports, "__esModule", { value: true });
var fs_1 = __importDefault(require("fs"));
function initTables() {
if (!fs_1.default.existsSync("./settings.json")) {
fs_1.default.writeFileSync("./settings.json", "{ }");
}
if (!fs_1.default.existsSync("./devices.json")) {
fs_1.default.writeFileSync("./devices.json", "[]");
}
if (!fs_1.default.existsSync("./prayerTimes.json")) {
fs_1.default.writeFileSync("./prayerTimes.json", "[]");
}
console.log("initialized");
}
exports.default = initTables;
//# sourceMappingURL=initTables.js.map

```

File: home.js

```

"use strict";
Object.defineProperty(exports, "__esModule", { value: true });
exports.default = "<!DOCTYPE html>\n<html>\n  <head>\n    <style>\n      .switch {\n        position: relative;\n        display: inline-block;\n        width: 60px;\n        height: 34px;\n        transform: scale(0.8);\n      }\n      .switch input {\n        opacity: 0;\n        width: 0;\n        height: 0;\n      }\n      .slider {\n        position: absolute;\n        cursor: pointer;\n        top: 0;\n        left: 0;\n        right: 0;\n        bottom: 0;\n        background-color: #ccc;\n        -webkit-transition: 0.4s;\n        transition: 0.4s;\n      }\n      .slider:before {\n        position: absolute;\n        content: \"\";\n        height: 26px;\n        width: 26px;\n        left: 4px;\n        bottom: 4px;\n        background-color: white;\n        -webkit-transition: 0.4s;\n        transition: 0.4s;\n      }\n      input:checked + .slider {\n        background-color: #2196f3;\n      }\n      input:focus + .slider {\n        box-shadow: 0 0 1px #2196f3;\n      }\n      input:checked + .slider:before {\n        -webkit-transform: translateX(26px);\n        transform: translateX(26px);\n      }\n      -ms-transform: translateX(26px);\n      transform: translateX(26px);\n    }\n    /* Rounded sliders */\n    .slider.round {\n      border-radius: 34px;\n    }\n  </style>\n  </head>\n  <body>\n    <div>\n      <input type='checkbox' />\n    </div>\n  </body>\n</html>";

```

```

.slider.round:before {\n          border-radius: 50%;\n        }\n        body {\npadding: 20px;\n        }\n        .center {\n          text-align: center;\n        }\n.loader {\n          font-size: 10px;\n          margin: 50px auto;\n          text-indent: -9999em;\n          width: 11em;\n          height: 11em;\n          border-radius: 50%;\n          background: #ffffff;\n          background: -moz-linear-gradient(\n            left,\n            #ffffff 10%,\n            rgba(255, 255, 255, 0) 42%\n          );\n          background: -webkit-linear-gradient(\n            left,\n            #ffffff 10%,\n            rgba(255, 255, 255, 0) 42%\n          );\n          background: -o-linear-gradient(\n            left,\n            #ffffff 10%,\n            rgba(255, 255, 255, 0) 42%\n          );\n          background: -ms-linear-gradient(\n            left,\n            #ffffff 10%,\n            rgba(255, 255, 255, 0) 42%\n          );\n          background: linear-gradient(\n            to right,\n            #ffffff 10%,\n            rgba(255, 255, 255, 0) 42%\n          );\n          position: relative;\n          -webkit-animation: load3 1.4s infinite linear;\n          animation: load3 1.4s infinite linear;\n          -webkit-transform: translateZ(0);\n          -ms-transform: translateZ(0);\n          transform: translateZ(0);\n        }\n.loader:before {\n          width: 50%;\n          height: 50%;\n          background: #ffffff;\n          border-radius: 100% 0 0 0;\n          position: absolute;\n          top: 0;\n          left: 0;\n          content: \"\";\n        }\n.loader:after {\n          background: #0dc5c1;\n          width: 75%;\n          height: 75%;\n          border-radius: 50%;\n          content: \"\";\n          margin: auto;\n          position: absolute;\n          top: 0;\n          left: 0;\n          bottom: 0;\n          right: 0;\n        }\n@-webkit-keyframes load3 {\n          0% {\n            -webkit-transform: rotate(0deg);\n            transform: rotate(0deg);\n          }\n          100% {\n            -webkit-transform: rotate(360deg);\n            transform: rotate(360deg);\n          }\n}\n@keyframes load3 {\n          0% {\n            -webkit-transform: rotate(0deg);\n            transform: rotate(0deg);\n          }\n          100% {\n            -webkit-transform: rotate(360deg);\n            transform: rotate(360deg);\n          }\n}\n</style>\n  <title>Set Up Athan</title>\n  <link\n    rel=\"stylesheet\"\n    href=\"https://fonts.googleapis.com/icon?family=Material+Icons\"\n    />\n  <link\n    rel=\"stylesheet\"\n    href=\"https://code.getmdl.io/1.3.0/material.indigo-pink.min.css\"\n    />\n  <script\n    defer src=\"https://code.getmdl.io/1.3.0/material.min.js\"\n    ></script>\n  <script\n    src=\"https://unpkg.com/react@18/umd/react.production.min.js\"\n    crossorigin\n    ></script>\n  <script\n    src=\"https://unpkg.com/react-dom@18/umd/react-dom.production.min.js\"\n    crossorigin\n    ></script>\n  <script\n    src=\"https://unpkg.com/babel-standalone@6/babel.min.js\"\n    ></script>\n  <script>\nfunction postApi(url, body) {\n  return fetch(url, {\n    method: \"POST\", \n    headers: {\n      \"Content-Type\": \"application/json\", \n    }, \n    body: JSON.stringify(body), \n  }).then((response) => response.json());\n}\n</script>\n</head>\n<body>\n  <div\n    id=\"root\"\n    ></div>\n  <script type=\"text/babel\"\n    >\n    const allPrayers = [\"fajr\", \"dhuhr\", \"asr\", \"maghrib\", \"isha\"];\n    const useState = React.useState;\n    const useEffect = React.useEffect;\n    function useApi(url) {\n      const [data, setData] = useState(null);\n      const [error, setError] = useState(null);\n      const [loading, setLoading] = useState(false);\n      const refetch = React.useCallback(() => {\n        setLoading(true);\n        fetch(url)\n          .then((response) => response.json())\n          .then((data) => {\n            setData(data);\n            setLoading(false);\n          })\n          .catch((error) => {\n            setError(error);\n            setLoading(false);\n          });\n      }, [url]);\n      useEffect(() => {\n        refetch();\n      }, [refetch]);\n      return { data, error, loading, refetch };\n    }\n    function App() {\n      return (\n        <div>\n          <h2\n            className=\"center\"\n            >Manage Athan Settings</h2>\n          <CityInput\n            />\n        </div>\n      );\n    }\n  </script>\n</body>\n</html>

```



```

<AllDevices />\n                </div>\n                );\n                }\n\n                function PrayerTimes()\n{\n    const { data, loading, error, refetch } = useApi("/api/prayertimes");\n    if (loading) {\n        return <div className="loader"></div>;\n    }\n    if (error) {\n        return <div>Error: {error.message}</div>;\n    }\n    if (!data) {\n        return <div>No data</div>;\n    }\n    return (\n        <div>\n            <p className="center">Prayer Times:</p>\n            {Object.keys(data).map((p) => (\n                <p>\n                    {p}: {new\nDate(data[p]).toLocaleTimeString()}\n                </p>\n            ))}\n        </div>\n    );\n    }\n\n    function CityInput() {\n        const { data,\nerror, loading, refetch } = useApi("/api/settings/list");\n        const [city,\nsetCity] = useState("");\n        useEffect(() => {\n            if (data && data.city)\n{\n                setCity(data.city);\n            }\n        }, [data]);\n        function\nsave() {\n            postApi("/api/settings/update/city", {\n                value: city,\n            }).then(() => {\n                refetch();\n            });\n        }\n        return (\n            <div className="center">\n                {data && data.city &&\n<PrayerTimes />}\n                <br />\n                <div className="center">\n                    City:{" "}\n                    <input\n                        type="text"\n                        value={city}\n                        onChange={(e) => setCity(e.target.value)}\n                    />\n                    <br />\n                    <button\n                        style={{\n                            backgroundColor: "#0dc5c1",\n                            color: "white",\n                            border: "none",\n                            padding: "10px 20px",\n                            borderRadius: "4px",\n                            marginTop: "10px",\n                            }}\n                        onClick={save}\n                    >\n                        Save City\n                    </button>\n                </div>\n            );\n        }\n\n        function ScanButton({ refetch }) {\n            const\n[scanning, setScanning] = useState(false);\n            function handleClick() {\n                setScanning(true);\n                fetch("/api/devices/scan")\n                .then((response) => response.json())\n                .then((data) => {\n                    setScanning(false);\n                    refetch();\n                });\n            }\n            if\n(scanning) {\n                return (\n                    <div>\n                        <p\n                            className="center">Scanning</p>\n                        <div className="loader\n                            center">Scanning...</div>\n                    </div>\n                );\n            }\n            return (\n                <button\n                    onClick={handleClick}\n                    style={{\n                        fontSize: 18,\n                        background: "purple",\n                        color:\n                            "white",\n                        padding: 10,\n                        borderRadius: 20,\n                        marginBottom: 20,\n                        }}\n                >\n                    Scan for New Devices\n                </button>\n            );\n        }\n\n        function AllDevices() {\n            const {\n                data, error, loading, refetch } = useApi("/api/devices/list");\n            if (loading)\n{\n                return <div>Loading...</div>;\n            }\n            if (error) {\n                return <div>Error: {error.message}</div>;\n            }\n            if (!data) {\n                return <div>No devices found</div>;\n            }\n            return (\n                <div>\n                    <h3 className="center">Devices</h3>\n                    <ScanButton refetch={refetch} />\n                    {data && !data.length &&\n(\n                <div className="center">No devices found</div>\n            )}\n                </div>\n                {\n                    data.map((device) => (\n                        <Device\n                            {...device} key={device.name} refetch={refetch} />\n                    ))\n                }\n            </div>\n        );\n    }\n\n    function Device({\n        id,\n        name,\n        friendlyName,\n        volume,\n        enabled,\n        prayers,\n        refetch,\n    }) {\n        const [newPrayers, setNewPrayers] = useState(prayers);\n        const\n[newVolume, setNewVolume] = useState(volume);\n        const [newEnabled, setNewEnabled]\n= useState(enabled);\n        const changesMade =\n            newPrayers.toString() !==\nprayers.toString() ||\n            volume !== newVolume ||\n            newEnabled !==\nenabled;\n        function togglePrayer(prayer) {\n            const newerPrayers =\n[...newPrayers];\n            const index = newerPrayers.indexOf(prayer);\n            if

```

```

(index === -1) {\n                newerPrayers.push(prayer);\n            } else {\n                newerPrayers.splice(index, 1);\n            }\n            setNewPrayers(newerPrayers);\n        }\n        function save() {\n            postApi(`"/api/devices/update/" + id,\n            {\n                prayers: newerPrayers,\n                volume: newVolume,\n                enabled: newEnabled,\n            }).then(() => {\n                refetch();\n            });\n        }\n        return (\n            <div>\n                <div\n                    style={{\n                        border: `1px solid grey`,\n                        borderRadius: 20,\n                        padding: 20,\n                        maxWidth: `calc(90vw - 20px)`,\n                    }}\n                    >\n                        <p>\n                            {changesMade && (\n                                <p style={{ color: `red` }}>You have unsaved changes</p>\n                            )}\n                        </p>\n                        <h5>Friendly Name: {friendlyName}</h5>\n                        <p>Full Name: {name}</p>\n                        <h6>Enabled: </h6>\n                        <label\n                            class=`switch`>\n                            <input\n                                type=`checkbox`\n                                checked={newEnabled}\n                                onChange={() => setNewEnabled(!newEnabled)}\n                            />\n                            <span class=`slider\n                                round`></span>\n                        </label>\n                        <br />\n                        <h6>Volume: </h6>\n                        <input\n                            type=`number`\n                            min=`0`\n                            max=`1`\n                            pattern=`^[0-1].[0-9]+`\n                            style={{ width: 50 }}\n                            step=`0.1`\n                            value={newVolume}\n                            onChange={(e) => setNewVolume(e.target.value)}\n                        />\n                        <br />\n                        <p>Prayers:</p>\n                        <p>Select which prayers you want to be played on this device</p>\n                        {allPrayers.map((p) => (\n                            <div key={p}>\n                                <input\n                                    type=`checkbox`\n                                    checked={newPrayers.includes(p)}\n                                    onChange={() => togglePrayer(p)}\n                                />{\n                                    ` `}\n                                {p}\n                            </div>\n                        )}}\n                        <br />\n                        {changesMade && (\n                            <button\n                                style={{\n                                    backgroundColor: `#0dc5c1`,\n                                    border: `none`,\n                                    padding: `10px 20px`,\n                                    borderRadius: `4px`,\n                                    cursor: `pointer`,\n                                    marginTop: `10px`,\n                                }}\n                                >\n                                    Save Changes\n                                </button>\n                            )}\n                        </div>\n                    <br />\n                    <hr />\n                </div>\n            );\n        }\n        </script>\n    </body>\n</html>\n";\n    // # sourceMappingURL=home.js.map

```

File: index.js

```

"use strict";
var __awaiter = (this && this.__awaiter) || function (thisArg, _arguments, P, generator) {
    function adopt(value) { return value instanceof P ? value : new P(function (resolve) {
        resolve(value);
    }); }
    return new (P || (P = Promise))(function (resolve, reject) {
        function fulfilled(value) { try { step(generator.next(value)); } catch (e) { reject(e); } }
        function rejected(value) { try { step(generator["throw"](value)); } catch (e) { reject(e); } }
        function step(result) {
            result.done ? resolve(result.value) : adopt(result.value).then(fulfilled, rejected);
        }
        step((generator = generator.apply(thisArg, _arguments || [])).next());
    });
};

```

```

};
var __generator = (this && this.__generator) || function (thisArg, body) {
var _ = { label: 0, sent: function() { if (t[0] & 1) throw t[1]; return t[1]; }, trys:
[], ops: [] }, f, y, t, g;
return g = { next: verb(0), "throw": verb(1), "return": verb(2) }, typeof Symbol ===
"function" && (g[Symbol.iterator] = function() { return this; }), g;
function verb(n) { return function (v) { return step([n, v]); }; }
function step(op) {
if (f) throw new TypeError("Generator is already executing.");
while (_) try {
if (f = 1, y && (t = op[0] & 2 ? y["return"] : op[0] ? y["throw"] || ((t = y["return"])
&& t.call(y), 0) : y.next) && !(t = t.call(y, op[1])).done) return t;
if (y = 0, t) op = [op[0] & 2, t.value];
switch (op[0]) {
case 0: case 1: t = op; break;
case 4: _.label++; return { value: op[1], done: false };
case 5: _.label++; y = op[1]; op = [0]; continue;
case 7: op = _.ops.pop(); _.trys.pop(); continue;
default:
if (!(t = _.trys, t = t.length > 0 && t[t.length - 1]) && (op[0] === 6 || op[0] === 2))
{ _ = 0; continue; }
if (op[0] === 3 && (!t || (op[1] > t[0] && op[1] < t[3]))) { _.label = op[1]; break; }
if (op[0] === 6 && _.label < t[1]) { _.label = t[1]; t = op; break; }
if (t && _.label < t[2]) { _.label = t[2]; _.ops.push(op); break; }
if (t[2]) _.ops.pop();
_.trys.pop(); continue;
}
op = body.call(thisArg, _);
} catch (e) { op = [6, e]; y = 0; } finally { f = t = 0; }
if (op[0] & 5) throw op[1]; return { value: op[0] ? op[1] : void 0, done: true };
}
};
var __importDefault = (this && this.__importDefault) || function (mod) {
return (mod && mod.__esModule) ? mod : { "default": mod };
};
Object.defineProperty(exports, "__esModule", { value: true });
var node_cluster_1 = __importDefault(require("node:cluster"));
var runApp_1 = __importDefault(require("./runApp"));
var sleep = function (ms) { return new Promise(function (res) { return setTimeout(res,
ms); }); };
if (node_cluster_1.default.isPrimary) {
node_cluster_1.default.fork();
// If for some reason the app shuts down, restart it
// Wait 20s to prevent an infinite restart that kills the pi
node_cluster_1.default.on("exit", function (workerId, code, signal) { return
__awaiter(void 0, void 0, void 0, function () {
return __generator(this, function (_a) {
switch (_a.label) {
case 0:
console.log("Worker ".concat(workerId, " died with code ").concat(code, " and signal
").concat(signal));
return [4 /*yield*/, sleep(20 * 1000)];
case 1:
_a.sent();

```

```
node_cluster_1.default.fork();
return [2 /*return*/];
}
});
}); });
node_cluster_1.default.on("online", function (worker) {
console.log("Worker ".concat(worker.process.pid, " is online"));
});
}
else {
(0, runApp_1.default)();
}
}
//# sourceMappingURL=index.js.map
```

Repository: auto-calendar-sharing

File: README.md

This is a [Next.js](https://nextjs.org/) project bootstrapped with [`create-next-app`](https://github.com/vercel/next.js/tree/canary/packages/create-next-app).

Getting Started

First, run the development server:

```
``bash
npm run dev
# or
yarn dev
``
```

Open http://localhost:3000 with your browser to see the result.

You can start editing the page by modifying `pages/index.tsx`. The page auto-updates as you edit the file.

[API routes](https://nextjs.org/docs/api-routes/introduction) can be accessed on http://localhost:3000/api/hello. This endpoint can be edited in `pages/api/hello.ts`.

The `pages/api` directory is mapped to `/api/*`. Files in this directory are treated as [API routes](https://nextjs.org/docs/api-routes/introduction) instead of React pages.

Learn More

To learn more about Next.js, take a look at the following resources:

- [Next.js Documentation](https://nextjs.org/docs) - learn about Next.js features and API.
- [Learn Next.js](https://nextjs.org/learn) - an interactive Next.js tutorial.

You can check out [the Next.js GitHub repository](https://github.com/vercel/next.js/) - your feedback and contributions are welcome!

Deploy on Vercel

The easiest way to deploy your Next.js app is to use the [Vercel Platform](https://vercel.com/new?utm_medium=default-template&filter=next.js&utm_source=create-next-app&utm_campaign=create-next-app-readme) from the creators of Next.js.

Check out our [Next.js deployment documentation](https://nextjs.org/docs/deployment) for more details.

File: 01_create-user.ts

```
import { DataTypes, QueryInterface } from "sequelize";
import { MigrationFn } from "umzug";

const up: MigrationFn<QueryInterface> = async function ({
  context: queryInterface,
}) {
  await queryInterface.createTable("User", {
    id: {
      type: DataTypes.INTEGER,
      allowNull: false,
      primaryKey: true,
      autoIncrement: true,
    },
    firstName: {
      type: DataTypes.STRING,
      allowNull: false,
    },
    lastName: {
      type: DataTypes.STRING,
      allowNull: false,
    },
    createdAt: {
      type: DataTypes.DATE,
      allowNull: false,
    },
    updatedAt: {
      type: DataTypes.DATE,
      allowNull: false,
    },
    deletedAt: {
      type: DataTypes.DATE,
      allowNull: true,
    },
  });
};

const down: MigrationFn<QueryInterface> = async function ({
  context: queryInterface,
}) {
  await queryInterface.dropTable("User");
};

export { up, down };
```

File: 02_add-email-to-user.ts

```
import { DataTypes, QueryInterface } from "sequelize";
import { MigrationFn } from "umzug";

const up: MigrationFn<QueryInterface> = async function ({
```

```

context: queryInterface,
})) {
await queryInterface.addColumn("User", "email", {
type: DataTypes.STRING,
allowNull: false,
});
};

const down: MigrationFn<QueryInterface> = async function ({
context: queryInterface,
})) {
await queryInterface.removeColumn("User", "email");
};

export { up, down };

```

File: User.ts

```

import { gql } from "apollo-server-micro";
export default gql`
type User {
id: Int!
email: EmailAddress!
firstName: String!
lastName: String!
createdAt: DateTime!
updatedAt: DateTime!
}
`;

```

File: sequelizeize.ts

```

import path from "path";
import { Options, Sequelize } from "sequelize";

const sqlitePath = path.resolve(__dirname, "../app.db");

const logsDisabled = process.env.SEQUELIZE_NO_LOG === "true";

const sequelize_url: string = process.env.SEQUELIZE_URL || "";

const url: string = sequelize_url || `sqlite://${sqlitePath}`;

const env: string = process.env.NODE_ENV || "development";

const configMap: { [key: string]: Options } = {
development: {
dialect: sequelize_url ? undefined : "sqlite",
storage: sequelize_url ? undefined : "app.db",
define: {

```

```
charset: "utf8",
collate: "utf8_unicode_ci",
},
ssl: true,
native: true,
logging: logsDisabled ? false : console.log,
},
production: {
pool: {
max: 5,
min: 0,
acquire: 30000,
idle: 10000,
},
define: {
charset: "utf8",
collate: "utf8_unicode_ci",
},
native: true,
ssl: true,
logging: false,
},
};
```

```
const config = configMap[env];
```

```
const sequelize = new Sequelize(url, config);
```

```
export default sequelize;
```

File: umzug.ts

```
import path from "path";
import { SequelizeStorage, Umzug } from "umzug";
import sequelize from "../sequelize";

const migrationsPath = path.resolve(__dirname, "migrations");

// Umzug is responsible for running Migrations
const umzug = new Umzug({
  migrations: {
    glob: migrationsPath + "/*.ts",
  },
  context: sequelize.getQueryInterface(),
  storage: new SequelizeStorage({ sequelize }),
  logger: console,
});

export default umzug;
```


File: index.ts

```
import Mutation from "../Mutation";
import Query from "../Query";
import scalars from "../scalars";
import User from "../User";

const typeDefs = [Mutation, User, Query, scalars];

export default typeDefs;
```

File: createUser.ts

```
import User from "../../database/models/User";

type CreateUserArgs = {
  email: string;
  firstName: string;
  lastName: string;
};

const createUser = (
  parent: null,
  { email, firstName, lastName }: CreateUserArgs
): Promise<User> => User.create({ email, firstName, lastName });

export default createUser;
```

File: userById.ts

```
import User from "../../database/models/User";

type UserByIdArgs = {
  id: number;
};

const userById = (parent: null, { id }: UserByIdArgs): Promise<User | null> =>
  User.findOne({ where: { id } });

export default userById;
```

File: Mutation.ts

```
import { gql } from "apollo-server-micro";

export default gql`
type Mutation {
  createUser(
```

```
email: EmailAddress!  
firstName: NonEmptyString!  
lastName: NonEmptyString!  
): User!  
}  
`;
```

File: Query.ts

```
import { gql } from "apollo-server-micro";  
  
export default gql`  
type Query {  
  userById(id: Int!): User  
}  
`;
```

File: scalars.ts

```
import { gql } from "apollo-server-micro";  
  
export default gql`  
# Defined in graphql-scalars  
scalar Date  
scalar Time  
scalar DateTime  
scalar Timestamp  
scalar UtcOffset  
scalar Duration  
scalar ISO8601Duration  
scalar LocalDate  
scalar LocalTime  
scalar LocalEndTime  
scalar EmailAddress  
scalar NegativeFloat  
scalar NegativeInt  
scalar NonEmptyString  
scalar NonNegativeFloat  
scalar NonNegativeInt  
scalar NonPositiveFloat  
scalar NonPositiveInt  
scalar PhoneNumber  
scalar PositiveFloat  
scalar PositiveInt  
scalar PostalCode  
scalar UnsignedFloat  
scalar UnsignedInt  
scalar URL  
scalar BigInt  
scalar Long
```

```
scalar Byte
scalar UUID
scalar GUID
scalar Hexadecimal
scalar HexColorCode
scalar HSL
scalar HSLA
scalar IPv4
scalar IPv6
scalar ISBN
scalar JWT
scalar Latitude
scalar Longitude
scalar MAC
scalar Port
scalar RGB
scalar RGBA
scalar SafeInt
scalar USCurrency
scalar Currency
scalar JSON
scalar JSONObject
scalar IBAN
scalar ObjectID
scalar Void
scalar DID
scalar CountryCode
`;
```

Repository: auto-prettify

File: README.md

quicker-picker-upper

Template for quick NodeJS prototyping and development

```
[[code style:prettier](https://img.shields.io/badge/code_style-prettier-ff69b4.svg?style=flat-square)](https://github.com/prettier/prettier)
```

Quick Start

1. Instal NodeJS on your computer if you haven't done so already:
<https://nodejs.org/en/download/>
2. Click the "Use this template" button on GitHub, right of the "Clone or download" button and give your repository a name.
3. Clone your new repository to your computer.
4. `cd` into your repository's directory and install dependencies:

```
```shell
npm install
```
```

5. That's it, you can start developing!

- If you want your app to automatically restart upon file changes, run it with the following command:

```
```shell
npm run dev
```
```

Why use it?

- [x] Has the things you'll need, less time setting up
- Rolling sessions, connected to the database, with signed cookies
- Simple, modular routing structure
- Body and cookie parsers
- [x] Stop worrying about configuring a database
- Uses [Sequelize ORM](<https://sequelize.org/>).
- Define your schema once and have it work with `mysql`, `postgres`, `sqlite`, or `mariadb` out of the box
- The app will automatically create a sqlite3 database with the name `app.db` if you don't provide a database
- [x] Continuous integration.
- Encourages unit-testing
- GitHub workflow, with caching, already set up
- Tests your code, in parallel on NodeJS 10.x and 12.x, as well as with every database type, on every commit or pull request
- [x] [Prettier](<https://prettier.io/>)-ready
- Automatically format your code with `npm run prettier`

- Spend more time writing code and less time worrying about formatting
- [x] Production Ready
- Creates clusters to distribute load across cpu cores
- Instantly revives child processes if they die. Your entire app won't go offline if something crashes in production.
- Pooling of database connections allows for quicker access to data
- [x] PaaS Ready (Platform as a Service)
- Instantly deploy on Heroku or AWS Elastic Beanstalk with ease
- Most configuration happens using environment variables

Databases

Define your schema using [Sequelize CLI](https://github.com/sequelize/cli)

Have a database already? You can use it with the app by setting the `SEQUELIZE_URL` environment variable. You can do this in a .env file like so:

```
```dotenv
SEQUELIZE_URL=mysql://user:password@localhost/my_db
```
```

Testing

Testing happens with the [mocha](https://mochajs.org/) and [chai](https://www.chaijs.com/) libraries.

By default, mocha will look for tests inside of the `test/` directory. You can alter the mocha configuration in the `package.json` file/